MOLE

MICROCONTROLLER DEVELOPMENT SUPPORT

HPC16083/HPC16043/HPC16003 USER'S MANUAL



Mational Semiconductor Corporation

		, a
		(

Customer Order Number 424410897-001 NSC Publication Number 424410897-001A July 1987

$HPC^{\text{\tiny TM}}$

HPC16083/HPC16043/HPC16003 User's Manual

 1987 National Semiconductor Corporation 2900 Semiconductor Drive
 P.O. Box 58090
 Santa Clara, California 95052-8090

REVISION RECORD

REVISION RELEASE DATE SUMMARY OF CHANGES

A 07/87 First Release.
HPC16083/HPC16043/HPC16003
User's Manual
NSC Publicaton Number 424410897-001.

PREFACE

This manual describes the hardware and software characteristics of the HPC16083, HPC16043 and HPC16003 microcontrollers. The HPC16083 has an 8-Kbyte ROM while the HPC16043 has a 4-Kbyte ROM and the HPC16003 is ROMless. Throughout the manual, the term "HPC16083" is used to refer to all three parts. The HPC16043 and HPC16003 are identical to the HPC16083 except for ROM size and a few minor details. The memory maps and differences of the HPC16043 and HPC16083 with respect to the HPC16083 are shown in Appendix G and H.

Because the HPC16083, HPC16043 and HPC16003 are members of the HPC family of High Performance microControllers and contain the same CPU core, the core hardware and software features described in this manual are the same as on other HPC family members. The assembler syntax and some instruction mnemonics shown in this manual are new. Appendix F shows the differences from the old assembler syntax and instruction mnemonics.

The HPC16083, HPC16043 and HPC16003 devices and the information contained within this manual supercede the HPC16040 and HPC16030 and the older Hardware and Programmer's manuals. The new features of the HPC16083, HPC16043 and HPC16003 are transparent additions and do no affect code or hardware originally intended for the HPC16040.

The information contained in this manual is for reference only and is subject to change without notice.

No part of this document may be reproduced in any form or by any means without the prior written consent of National Semiconductor Corporation.

HPC, MICROWIRE/PLUS and COPS are trademarks of National Semiconductor Corporation.

CONTENTS

Chapter 1	GENERAL DESCRIPTION	-1
		-1 -1
Chapter 2	ARCHITECTURAL FEATURES	2-1
	2.1 INTRODUCTION	2-1
	2.2 MEMORY-MAPPED VON NEUMANN ARCHITECTURE 2	2-1
	2.3 CARRY USE AND REPORTING	2-2
	2.4.1 Addressing Bytes and Words	2-2 2-2 2-3 2-5
,	2.5 CONDITIONAL INSTRUCTIONS AND SKIPPING 2	2-6
Chapter 3	PROGRAMMING MODEL	3-1
	3.1 INTRODUCTION	3–1
Chapter 4	ADDRESSING MODES	1-1
	4.1 INTRODUCTION	1-1
	4.2.1 B Register Indirect	1-4 1-4 1-5 1-5
	4.3.1 Immediate	1-6 1-6 1-6 1-6
	4.4.1 X Register Indirect with Auto-Increment/Decrement 4 4.4.2 B Register Indirect with Auto-Increment/Decrement and	4-7 4-7
		4-7 4-9
Chapter 5	DETAILED INSTRUCTION DEFINITIONS	5-1
	5.1 INTRODUCTION	5-1
	5.2 ORGANIZATION AND NOTATIONS	5-1
Chapter 6	FUNCTIONAL PIN DESCRIPTION	6-1
	6.1 HPC16083 PIN DESCRIPTION	5-1
Chapter 7	HARDWARE CONNECTIONS	7-1

	7.1	POWER AND GROUNDING
	7.2	CLOCKING
	7.3	RESET 7-3
Chapter 8	MEM	IORY ORGANIZATION AND REGISTER SET 8-1
	8.1	INTRODUCTION
	8.2	MEMORY MAP
	8.3	CONTROL REGISTERS 8-1
	8.4	DEDICATED REGISTERS 8-1
	8.5	PROCESSOR STATUS WORD (PSW) 8-5
Chapter 9	PORT	STRUCTURES AND OPERATION 9-1
	9.1	INTRODUCTION
	9.2	PORT A
	9.3	PORT B
	9.4	PORT I
	9.5	PORT D
	9.6	PORT P
Chapter 10	BUS	FUNCTIONS AND CONFIGURATIONS 10-1
	10.1	BUS MODES AND SELECTION 10-1 10.1.1 Single Chip Mode vs. Expanded Mode 10-1 10.1.2 Normal Mode vs. ROMless Mode 10-3 10.1.3 8-Bit vs. 16-Bit Mode 10-4
	10.2	SYSTEM CONFIGURATION TABLES 10-6
	10.3	BUS OPERATION
	10.4	WAIT STATES
	10.5	REQUESTING WAIT STATES 10-13
	10.6	ADDRESSING BYTE/WORD EXTERNAL MEMORIES 10-18
	10.7	STATUS PINS ST1 AND ST2
	10.8	SYSTEM BUS BEHAVIOR 10-21 10.8.1 Single Chip Normal Mode 10-21 10.8.2 Expanded Normal 8-bit 10-23 10.8.3 Expanded Normal 16-bit 10-23 10.8.4 Single Chip ROMless 8-bit 10-24 10.8.5 Single Chip ROMless 16-bit 10-25 10.8.6 Expanded ROMless 8-bit 10-25 10.8.7 Expanded ROMless 16-bit 10-26
	10.9	PROGRAMMING CONSIDERATIONS IN 8-BIT MODES 10-26
Chapter 11	SHAF	RED MEMORY SUPPORT (DMA)
	11.1	INTRODUCTION
	11.2	TIMING SEQUENCES

	11.3	PIN CONFIGURATION
Chapter 12	RESE	TT STATE
	12.1	INTRODUCTION
Chapter 13	WAT	CCHDOG LOGIC
	13.1	INTRODUCTION
	13.2	POTENTIALLY INFINITE LOOPS 13-1
	13.3	ILLEGAL ADDRESSES
Chapter 14	POW	ER SAVE MODES OF OPERATION 14-1
	14.1	INTRODUCTION
	14.2	HALT MODE
	14.3	IDLE MODE
Chapter 15	INTE	RRUPTS
	15.1	INTRODUCTION
	15.2	INTERRUPT PROCESSING
	15.3	INTERRUPT CONTROL REGISTERS 15-3
	15.4	INTERRUPT LATENCY
	15.5	INTERRUPT CONTROL REGISTER MAPS 15-5
		15.5.1 ENIR — Interrupt Enable Register
		15.5.3 EICON — EI Configuration Register 15-6
		15.5.4 IRPD — Interrupt Pending Register 15-8
Chapter 16		ERS
	16.1	INTRODUCTION
	16.2	TIMER OPERATIONS
		16.2.2 Timer T1
		16.2.3 Timers T2 and T3
	460	16.2.4 PWM Timers: T4—T7
		TIMER SECTION REGISTERS
	16.4	TIMER SETUP AND CONFIGURATION
	16.5	TIMER APPLICATIONS
Charter 17	16.6	ROWIRE/PLUS
Chapter 17		INTRODUCTION
	17.1 17.2	REGISTERS
		INITIALIZATION
Chapter 19		T
CHARLET 10		

	18.1	INTRODUCTION	18-1
	18.2	ASSOCIATED I/O PINS	18-1
	18.3	UART OPERATION	18-1
	18.4	FRAME FORMATS SUPPORTED	18-3
	18.5	ERROR FLAGS	18-4
	18.6	INTERRUPTS	18-6
	18.7	CLOCKING	18-7
	18.8	WAKE-UP MODE	18-7
	18.9	UART CONTROL REGISTERS	18-9 18-9
Chapter 19	UNIV	VERSAL PERIPHERAL INTERFACE (UPI)	19-1
	19.1	INTRODUCTION	19-1
	19.2	INTERNAL ORGANIZATION	19-1
	19.3	UPI CONTROL REGISTER	19-4
951	19.4	PIN FUNCTION DETAILS	19-5 19-5 19-6
	19.5	HANDSHAKING	19-6
	19.6	UPI MODE SETUP	19-7
Appendix A	REG	ISTERS AND MEMORY MAP	A-1
	A.1	HPC PROGRAMMING MODEL	A-1
	A.2	HPC16083 MEMORY MAP	A-2
	A.3	HPC16083 REGISTER BIT MAPS	A-4
Appendix B	INST	RUCTION SET SUMMARY	B-1
Appendix C	INST	RUCTION SET ENCODING, LENGTH AND TIMING	C-1
	C.1	INSTRUCTION STRUCTURE	C-1
	C.2	ADDRESSING DIRECTIVE PREFIXES AND RESULTING INSTRUCTION LENGTHS	C-2
	C.3	INSTRUCTION TABLE CONTENTS	C-3
	C.4	INSTRUCTION EXECUTION TIME	C-3
	C.5	INTERRUPT PROCESSING TIME	C-4
	C.6	SKIP PROCESSING TIME	C-4
	C.7	OPCODES AND LENGTHS (TWO-ADDRESS INSTRUCTIONS)	C-5
	C.8	CYCLES AND ACCESSES (TWO-ADDRESS INSTRUCTIONS)	C-6
	C.9	OPCODES AND LENGTHS (ONE-ADDRESS INSTRUCTIONS)	C-8
	C.10	CYCLES AND ACCESSES (ONE-ADDRESS INSTRUCTIONS)	C-10
	C.11	OPCODES AND LENGTHS (SPECIALIZED INSTRUCTIONS)	C-11

	C.12	CYCLES AND ACCESSES (SPECIALIZED INSTRUCTIONS) (C-15
	C.13	HPC OPCODE MAP	C-18
Appendix D	BASI	C ASSEMBLY LANGUAGE SYNTAX	D-1
	D.1	ASSEMBLER SYNTAX	D-1
	D.2	SUMMARY OF ASSEMBLER DIRECTIVES	D-3
Appendix E	ADD	RESSING MODES AND BINARY FORMATS	E-1
	E.1	INTRODUCTION	E-1
	E.2	ENCODING SCHEMES	E-1
	E.3	ONE-ADDRESS GENERAL MODES E.3.1 Register B Indirect E.3.2 Register X Indirect E.3.3 Direct E.3.4 Indirect E.3.5 Indexed	E-2 E-3 E-4 E-5 E-7 E-8
	E.4		E-9 E-9 E-10 E-14
Appendix F	REVI	SION TO ASSEMBLER SYNTAX	F-1
	F.1	INTRODUCTION	F-1
	F.2	ADDRESSING MODE DIFFERENCES	F-1
	F.3	INSTRUCTION MNEMONICS DIFFERENCES	F-2
$Appendix \ G$	ROM	LESS HPC16003	G-1
	G.1	INTRODUCTION	G-1
	G.2	HPC16003 RESTRICTIONS	G-1
Appendix H	DIFF	ERENCES IN THE HPC16043	H-1
	H.1	INTRODUCTION	H-1
	H.2	HPC16043 DIFFERENCES (WITH RESPECT TO THE HPC16083)	H-1
Appendix I	ASCI	I CHARACTER SET	I-1
FIGURES			
Figure 1-1.	HPC1	16083 Block Diagram	1-3
Figure 6-1.	Pinou	at Diagram for HPC16083: 68-Pin PLCC and LCC Packages	6-2
Figure 6-2.	Pinou	nt Diagram for HPC16083: 68-Pin PGA Package	6-3
Figure 7-1.	Powe	r Supply Connections	7-1
Figure 7-2.	On-C	Thip Clocking Circuit	7-2
Figure 7-3	Typic	cal Crystal Oscillator Connections	7-2

Figure 7-4.	External Clocking Connections
Figure 7-5.	Power-On Reset Circuit
Figure 9-1.	Structure of Port A
Figure 9-2.	Structure of Port B Pins B0, B1, B2, B5, B6 and B7 (Generic Structure) 9-5
Figure 9-3.	Structure of Port B Pins B3, B4, B8, B9, B13 and B14 (Timer Synchronous Pins)
Figure 9-4.	Structure of Port B Pins B10, B11, B12 and B15 (Pins with Bus Control Roles)
Figure 9-5.	Structure of Port I
Figure 9-6.	Structure of Port D
Figure 9-7.	Structure of Port P
Figure 10-1.	System Configuration: Single Chip Mode
Figure 10-2.	System Configuration: 16-Bit Modes, Gating Chip Selects 10-5
Figure 10-3.	System Configuration: 8-Bit Modes
Figure 10-4.	Bus Timing: Read Cycle; No Wait States
Figure 10-5.	Bus Timing: Write Cycle; No Wait States 10-11
Figure 10-6.	Bus Timing: Read Cycle; One Wait State
Figure 10-7.	Bus Timing: Write Cycle; One Wait State
Figure 10-8.	READY Signal Timing
Figure 10-9.	System Configuration: 16-Bit Modes, Gating Write Strobe 10-19
Figure 11-1.	Shared Memory Application
Figure 11-2.	HOLD / HLDA Timing: Entry into Hold States
Figure 11-3.	HOLD / HLDA Timing: Exit from Hold States
Figure 14-1.	HALT Timing: Using NMI to Exit HALT Mode
Figure 15-1.	Block Diagram of Interrupt Logic
Figure 16-1.	Timer TO, T1 and T8 With Four Input Capture Registers 16-3
Figure 16-2.	Timer Block Diagram: Timers T2 and T3 16-5
Figure 16-3.	PWM Timers (T4-T7) Block Diagram 16-6
Figure 17-1.	Example of a MICROWIRE Application
Figure 17-2.	MICROWIRE/PLUS Interface Block Diagram 17-3
Figure 17-3.	MICROWIRE/PLUS Transfer
Figure 18-1.	UART Block Diagram
Figure 18-2.	Character Frame Formats
Figure 18-3.	Data Flow Between TSFT/RSFT and Other Registers 18-5
O	UART Interrupt Generation
•	UPI Block Diagram
	UPI Timing Diagrams

Figure 19-3.	Simple UPI Application
Figure 19-4.	Interrupt-Driven UPI Application: UPI Interface to Series 32000 System
Figure 19-5.	Interrupt-Driven UPI Application: UPI Interface to NSC800 19-10
Figure 19-6.	UPI Application Inserting WAIT States 19-11
Figure 19-7.	Host Wait State Insertion Sequence of Events (Reading) 19-12
Figure 19-8.	Host Wait State Insertion Sequence of Events (Writing) 19-13
TABLES	
TABLE 4-1.	GENERAL ADDRESSING MODES
TABLE 4-2.	INSTRUCTIONS SUPPORTING TWO ADDRESSES 4-3
TABLE 4-3.	INSTRUCTIONS SUPPORTING ONE ADDRESS 4-3
TABLE 8-1.	MEMORY MAP OF HPC16083
TABLE 8-2.	CONTROL REGISTERS
TABLE 10-1.	SYSTEM CONFIGURATION CHART 10-8
TABLE 10-2	MEMORY SPACES AND ACCESSIBILITY 10-9
TABLE 10-3	MEMORY ORGANIZATION/AVAILABILITY CHART 10-22
TABLE 15-1.	INTERRUPT ARBITRATION
TABLE 15-2	EICON REGISTER (BITS 1 AND 0)
TABLE 16-1	FUNCTIONS OF PINS B3 (T2IO) AND B4 (T3IO) 16-16
TABLE 16-2	TS0—TS3 PIN FUNCTIONS
TABLE 18-1	. UART CLOCK SOURCES FROM DIVBY REGISTER 18-8
TABLE G-1.	HPC16003 MEMORY MAP
TABLE H-1.	MEMORY MAP OF HPC16083

INDEX

		•

Chapter 1

GENERAL DESCRIPTION

1.1 ADVANCED SINGLE CHIP CMOS MICROCONTROLLER

The HPC16083 single chip microcontroller is a complete microcomputer containing all the necessary system timing, internal logic, ROM, RAM, and I/O for implementing dedicated control functions in a variety of applications. Every member of the HPC^{IM} family is designed in National's advanced microCMOS technology. This process allows implementation of the fast, flexible I/O control, efficient data manipulation, and high-speed computation that characterize the HPC family.

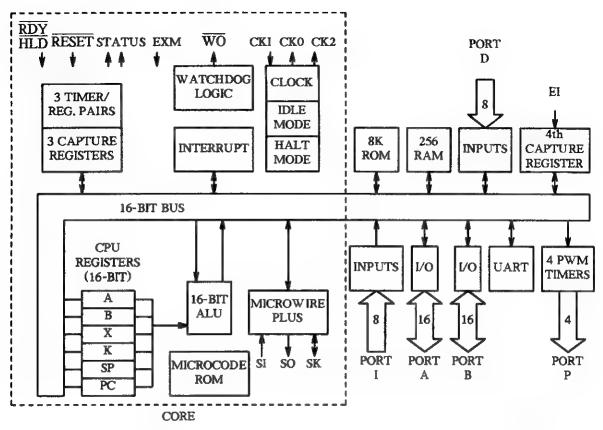
The HPC16083 is a complete microcomputer on a single chip. On-chip functions such as the UART, the nine 16-bit timers, and the MICROWIRE/PLUSTM interface provide a high level of system integration (see Figure 1-1).

The microCMOS process results in very low current drain and enables the user to select the optimal speed/power product for his system.

1.1.1 Features

- CMOS very low power with two power-save modes: HALT and IDLE
- FAST! 134 ns for register instructions when using 30.0 MHz clock
- 16-bit data bus, ALU, and registers
- Enhanced architecture using BOTH 16-bit words and 8-bit bytes
- Enhanced instruction set
- High code efficiency with single-byte Jumps, Calls and RAM access
- 16×16 Multiply and 32×16 Divide instructions
- 8K×8 on-chip ROM, 256×8 on-chip RAM
- Off-chip expansion to 64K (RAM and/or ROM; code and/or data)
- 16-bit Stack Pointer register
- Memory-Mapped Register Set: Core and I/O Registers
- 52 I/O lines
- Synchronous serial I/O capability using MICROWIRE/PLUS
- Full-duplex UART with programmable transmit & receive baud rates Full-duplex UART with programmable transmit & receive baud rates Eight vectored interrupt sources
- Nine 16-bit timers with up to ten outputs
- Timer T0 has up to three Input Capture registers

- Processor monitored by Watchdog feature
- Low-power CMOS with ultra low power HALT mode



GL-01-0-U

Figure 1-1. HPC16083 Block Diagram

		_
		_
		$\overline{}$
		·
		$\overline{}$

Chapter 2

ARCHITECTURAL FEATURES

2.1 INTRODUCTION

The architectural philosophy of the HPC family is drawn from several sources; ideas for code efficiency taken from the COP400TM family of microcontrollers accompany other features that are more typical of 8-bit and 16-bit (and even 32-bit) microprocessors. Thus, depending on one's background, one feature or another of the HPC is bound to be unfamiliar. This chapter will explain some of the more unique features of the HPC, and how they are meant to be used.

The topics covered in this chapter are:

- Memory-Mapped Von Neumann Architecture
- Carry Use and Reporting
- 8-/16-Bit Data Handling
- Conditional Instructions and Skipping

2.2 MEMORY-MAPPED VON NEUMANN ARCHITECTURE

In the HPC family, the instruction memory and data memory co-exist in the same addressing space. A program is allowed to access data in the program area (including the on-chip ROM), and it is allowed to place small routines into the on-chip RAM and execute them from there (with a significant gain in performance over off-chip memory). This scheme is referred to as Von Neumann memory architecture.

In addition, all software data registers, and I/O control and data registers, are accessible as bytes and/or words with memory-mapped addresses. This scheme is referred to as memory-mapping of the programming model and I/O.

The combination of Von Neumann and memory-mapping philosophies allow a very powerful instruction set to be encoded within the 8-bit opcodes of the HPC, by eliminating the need for many specialized types of instructions. Some examples:

X A, K ; Exchange Accumulator with Register K

This instruction is simply one case of the X instruction, which exchanges the contents of the accumulator with those of any memory location. Since the K register has a memory-mapped address (OOCA Hex), this instruction simply references it as memory, using an 8-bit addressing field within the instruction. Dedicated register-to-register instructions are unnecessary, and therefore not present, in the HPC architecture.

SBIT 7, SIO ; Echo Character with Acknowledge Bit Set

This instruction performs an I/O operation, using the general purpose Set Bit (SBIT) instruction. The SIO register, which contains the last byte received from the MICROWIRE/PLUS^{IM} interface (an ASCII character in this case), is modified by setting its top bit. This action triggers transmission of the new value. Thus, this single instruction echoes the last character received,

having set its top bit as an acknowledgment. Dedicated instructions for I/O operations are unnecessary, and therefore not present, in the HPC architecture.

2.3 CARRY USE AND REPORTING

The C Bit (bit 5 of the PSW register) is used as a direct carry input to the Arithmetic Logic Unit (ALU) during the instructions ADC, SUBC, DADC and DSUBC. It also receives the direct carry output from the ALU upon execution of these instructions, and also on executing the ADD and ADDS instructions, which do not use the C bit as an input.

Because the C bit represents a direct binary carry, its function differs between addition and subtraction operations.

In an addition that accepts a carry in, the exact sum of the two operands is returned if the C bit is a zero. If it is a one, the result is greater by one than the exact sum. The carry seen in the C bit after the addition is a one to indicate a carry condition (unsigned overflow), and a zero otherwise.

Both of the subtraction instructions (SUBC and DSUBC) accept a carry in. This carry, however, must be initialized to a one in order to calculate the exact difference of the two operands; if the C bit is a zero, the result is less by one than the exact difference. The carry seen in the C bit after subtraction is a zero to indicate an unsigned overflow (borrow), and a one otherwise.

2.4 8-/16-BIT DATA HANDLING

The HPC was designed to handle 8-bit and 16-bit data with equal ease. Three outgrowths of this philosophy are detailed here:

- Addressing Bytes and Words
- Mixing Data Sizes
- Examples

2.4.1 Addressing Bytes and Words

Memory in the HPC family is addressed in units of bytes, each byte consisting of eight bits. A word is a 16-bit value, consisting of two contiguous bytes. When specifying the address of a word, an even address must always be used, this being the address of the least-significant byte (LSB) of the word. The most-significant byte (MSB) of the word has the odd address that is one greater than the address of the LSB. It is not allowed in the HPC architecture to place a word in memory such that its LSB is at an odd address.

Depending on the mode of operation selected at Reset, 16-bit accesses may be illegal entirely outside of the addressing range 0000 — 01FF (on-chip RAM and registers) and E000-FFFF (on-chip ROM). See Section 10.9 for a detailed explanation of the resulting restrictions.

All 16-bit registers on the HPC chip may be accessed as bytes as well as words. It is therefore possible, for example, to increment only the low-order half of the B register, or to modify only

the high-order byte of the Port A I/O data register, by making appropriate references to their memory-mapped addresses.

2.4.2 Mixing Data Sizes

The HPC performs all arithmetic internally as 16-bit calculations. Unlike most other microcomputers, however, the HPC allows calculations to be performed directly on values that are of differing sizes; for example, one can add a one-byte value directly to a word, without performing a conversion step first. This is possible because, rather than requiring the programmer to perform the conversion, the HPC automatically performs zero-extension of 8-bit values to 16 bits before use (that is, it appends a byte of all zeroes as the top half of the 16-bit value). Word destinations receive the entire 16-bit result, regardless of the size of the source operand, and single-byte destinations receive only the least-significant eight bits of the 16-bit result.

The zero-extension scheme was selected for the HPC because, as a processing element for control applications, it primarily handles unsigned values; that is, values that are never interpreted as negative numbers (for example, addresses). Zero-extension is the correct scheme to use when the one-byte value is considered to be a positive number in the range 0-255 (decimal). For example, the one-byte value 255 (FF Hex) remains 255 (00FF Hex) if its original interpretation is unsigned.

If, however, a one-byte value is considered to be a two's-complement number (range -128 to +127), then a different scheme must be used for the transformation. If the value's sign bit (bit 7) is set, then the top half of the converted value must be all ones instead of zeroes. Since it is impossible for the HPC to know in advance what a value is intended to represent, it remains the programmer's responsibility to extend signed numbers. This, however, is not a difficult or inefficient process. It can be accomplished as follows (and note that the automatic zero-extension actually helps here):

LD TEMPWD, BVAL ; Load a temporary word from the byte ; value, with automatic zero-extension.

IFBIT 7, TEMPWD . B ; Check if value is negative (sign bit = 1).

DECSZ TEMPWH ; If so, decrement top byte (from 00 to FF Hex).

where BVAL is the signed (two's complement) value to be converted, TEMPWD is a temporary word (preferably somewhere in on-chip RAM for efficiency), and TEMPWH is the high-order byte of that word. Now the converted value TEMPWD contains the proper sign-extended value from the original BVAL variable. There is no reason, by the way, that the accumulator could not be used instead of the RAM location TEMPWD.

Note that 16-bit signed values need no conversion for use in addition and subtraction; if care is taken to keep signed values within appropriate limits, the correct result, by either signed or unsigned interpretation, is always returned.

Implications for the Accumulator

Both bytes of the Accumulator (Register A) are altered by default when it is a destination. This convention, however, can be bypassed by individually addressing the bytes of the accumulator using their memory-mapped addresses. For example, to increment only the low-order byte of the accumulator without allowing the carry (if any) to affect the high-order byte, one can state:

INC HC8.B ; C8 is address of Accumulator.

instead of the no-address instruction:

INC A

Implications for the Carry Flag

The fact that the entire accumulator is automatically altered does not imply that the Carry flag always indicates a Word carry. To provide for easy manipulation of 8-bit values in the Accumulator, the Carry flag is loaded during addition or subtraction from the carry out of the bit position (7 for bytes, 15 for words) that is appropriate for the sizes of the operands involved. The carry position is determined according to the size of one of the two operands, and the selection of which operand controls the carry position is determined from the addressing mode used to access them, as follows:

ADDRESSING MODE	CARRY POSITION DETERMINED BY SIZE OF
B Indirect	Source Operand
X Indirect	Source Operand
Direct	Source Operand
Indirect	Source Operand
Indexed	Source Operand
Immediate	Neither: Word carry always, except in ADDS instruction (Byte)
Direct-Direct	Destination Operand
Immediate-Direct	Destination Operand

These conventions recognize that the Accumulator is not used as a final destination for results. Eight-bit values are often manipulated there, requiring that an 8-bit carry be reported (and ignoring the top halves of the temporary results held there). On the other hand, when a result is returned directly to memory, what is most often required of the C bit is an indication of a carry beyond the bounds of the destination.

Thus, when a one-byte operand from memory is added to the Accumulator as the destination selected by an addressing mode, the entire Accumulator is affected but the operation is considered an 8-bit addition for purposes of generating a carry. However, if a byte is added directly to a word in memory, not only is the entire word modified, but the carry also reflects the result expected of a 16-bit addition.

If one wishes to bypass these conventions (e.g., to add a byte to the Accumulator with a 16-bit carry) it is always possible to use memory mapping to change the interpretation of the

operation. For example, if the accumulator is referenced as a word at its memory-mapped address (C8 Hex), adding a byte to it will generate a 16-bit rather than an 8-bit carry. Or, to add a byte to a word in memory and receive an 8-bit carry, one can add the byte to the low-order byte of the destination (although this does not propagate the carry through the entire word).

2.4.3 Examples

The first three examples illustrate cases where the Accumulator is the destination.

Example 1:

```
LD A,# H'EFFE; A \leftarrow EFFE Hex
ADD A,#4; A \leftarrow F002 Hex, and C \leftarrow 0
```

In this case no carry is generated, because the Accumulator is treated as a 16-bit destination for both carry and data when the source operand is immediate.

Example 2:

```
LD A,# H'EFFE; A \leftarrow EFFE Hex
LD 23.B,#4; Memory byte at 23 gets 4.
ADD A,23.B; A \leftarrow F002, and C \leftarrow 1
```

In this case the C bit reports a byte overflow, because the source operand is a byte. The entire accumulator is still affected.

Example 3:

```
LD A,#H'C823 ; A \leftarrow C823 \text{ Hex}
ADD A,#H'8900 ; A \leftarrow 5123 \text{ Hex}, and C \leftarrow 1
```

In this case the C bit reports a word overflow.

The following examples have a directly addressed memory location as the destination instead of the Accumulator. In this case the destination can be specified as either a byte or a word, and it is the destination that determines the carry position.

Let LOCB be a byte variable in memory, and LOCW be a word variable.

Example 4:

```
LD LOCB,#H'23 ; LOCB \leftarrow 23 Hex
ADD LOCB,#H'E0 ; LOCB \leftarrow 03 Hex, and C \leftarrow 1
```

In this case a byte carry is generated, since the destination is a byte.

Example 5:

```
LD LOCW,#H'56 ; LOCW \leftarrow 56
ADD LOCW,#H'FEC0 ; LOCW \leftarrow FF16 (Hex), and C \leftarrow 0
```

In this case the C bit is reset to show that no word carry was generated, even though a byte carry did occur.

Example 6:

LD LOCB,#H'45 ; LOCB \leftarrow 45 Hex LD LOCW,#H'00F5 ; LOCW \leftarrow 00F5 Hex ADD LOCW, LOCB ; LOCW \leftarrow 013A Hex, and C \leftarrow 1

In this case no carry is reported; no word carry was generated even though a byte carry did occur.

2.5 CONDITIONAL INSTRUCTIONS AND SKIPPING

Unlike most microprocessors, the HPC does not have conditional branching instructions. Instead, a skipping technique is implemented, which can be used to make any instruction conditional (not just branches). A set of instructions have the attribute that they will skip (not execute) the following instruction under some condition. This has many applications in controller environments; for example, the following block move loop, which contains only three one-byte instructions:

LOOP: LD A, [X+], B; Load A through X and increment X.

XS A, [B+], B; Exchange A with byte addressed by B,

increment B, and skip if B > K.

JP LOOP; Jump to LOOP (if not skipped).

It is not necessary for the following instruction to be any fixed number of bytes long; the HPC, by prefetching the first byte of the instruction, can determine its full length at the time it skips, and adjust the Program Counter accordingly. The only instruction that should not immediately follow a skipping instruction is a multiway branch (JID or JIDW), since it is always followed by a table rather than by code.

The instructions that skip are:

IFGT This instruction executes the following instruction if its first listed operand is greater than the second operand. The following instruction is skipped otherwise.

IFEQ This instruction executes the following instruction if its operands are equal. The following instruction is skipped otherwise.

IFBIT This instruction executes the following instruction if the specified bit is a one. The following instruction is skipped otherwise.

IFC This instruction executes the following instruction if the C bit in the PSW register is a one. The following instruction is skipped otherwise.

IFNC This instruction executes the following instruction if the C bit in the PSW register is a zero. The following instruction is skipped otherwise.

DECSZ This instruction decrements a value, and skips the following instruction if the result is zero.

LDS, XS These instructions, which use the specialized auto-increment/decrement addressing modes [B+] and [B-], will skip the following instruction if the B

register is modified such that it crosses the address boundary contained in the K register. That is, if the B register is incremented and becomes greater than K, or if it is decremented and becomes less than K, the following instruction is skipped.

RETSK This instruction performs a subroutine return, but unconditionally skips the first instruction encountered. This allows a subroutine to efficiently indicate an error or other exceptional condition as it returns.

In addition to skipping, another form of decision-making capability is provided in the HPC: multiway branching. The JID and JIDW instructions are provided for this purpose. They are followed by a table of bytes (JID) or words (JIDW) which contain PC-relative displacements to the various possible destinations of the jump. The appropriate table entry is selected by the contents of the Accumulator.

		(
		~

Chapter 3

PROGRAMMING MODEL

3.1 INTRODUCTION

The programming model for the HPC consists of six 16-bit registers and the Carry bit, which are described below. All registers have memory-mapped addresses, as listed, and their bytes may be addressed and accessed individually.

Accumulator (A) Register: at 00C8 Hex

The 16-bit A register is the default destination register for most instructions, although memory-to-memory addressing modes are also generally available.

Address (B) Register: at 00CC Hex

The 16-bit B register is the primary address register, used for register indirect addressing. In certain instructions it may be incremented or decremented automatically to move through memory tables and arrays. The B register is also compared with the K register in these instructions, and can trigger a skip to drop out of a loop when a limit is reached.

Address (X) Register: at 00CE Hex

The 16-bit X register may be used for indirect addressing in the same manner as the B register. Like the B register, it may also be auto-incremented or auto-decremented in certain instructions, but without a limit test. Using the X register in a loop in conjunction with the B register allows, for example, a memory-to-memory block move to be performed in a loop consisting of only three one-byte instructions.

Boundary Constant (K) Register: at 00CA Hex

The 16-bit K Register is used to set a limit on the auto-incrementing or auto-decrementing of the B Register. In the instructions that do this, a skip is triggered whenever the B Register is advanced beyond the address contained in the K Register.

Stack Pointer (SP) Register: at 00C4 Hex

The 16-bit SP register is the Stack Pointer, which addresses the stack. The SP register is incremented by two for each Push or Subroutine Call instruction, and decremented by two for each Pop or Return instruction. Interrupts also push a 16-bit return address on the stack. The SP register may be written as well as read using its memory-mapped address, and by doing so the stack may be placed anywhere in 16-bit memory and may be as deep as the available memory permits.

Program Counter (PC) Register: at 00C6 Hex

The 16-bit PC register addresses program memory throughout the 64K addressing range. It is a read-only register.

Carry (C) Bit: in PSW at 00C0 Hex

The Carry bit is contained within the PSW register, in bit position 5. It indicates an unsigned overflow or borrow after an addition or subtraction instruction is performed. Both byte and word carries may be selected to affect the Carry bit. Addition instructions set the Carry bit on unsigned overflow, and subtraction instructions clear the Carry bit on a borrow condition. In the instructions that are intended for multiple-precision addition or subtraction (ADC, SUBC, DADC and DSUBC), the Carry bit is also used as an input, to propagate a carry or borrow from a previous instruction. The MULT and DIV instructions always clear the Carry bit. The DIVD (Divide Double-Word) instruction clears it unless an overflow or divide-by-zero error is detected, in which case it is set. The Carry bit can be set or reset directly, using the "SC" and "RC" instructions, and so may also be used as a general purpose flag. The Carry bit is also used with the shifting instructions SHL, SHR, RLC and RRC as a destination and/or source of shifted bits. See Sections 2.3 and 2.4.2 for additional information on the Carry bit.

Chapter 4

ADDRESSING MODES

4.1 INTRODUCTION

The addressing modes of the HPC provide many powerful means of addressing operands. They fall into two categories general and specialized. The general addressing modes are those that are uniformly available on a large number of instructions. They are listed in Table 4-1. The specialized modes are available in a smaller number of instructions, and are described in Section 4.4.

Because this manual documents the HPC instruction set, and not the assembler, symbolic addressing expressions are not covered here. However, it should be noted that any addressing value in Table 4-1 of the form "address.B", "source.B", "dest.B", "address.W", "source.W" or "dest .W" can be replaced by a properly declared symbol.

An HPC instruction can have one or two operands, whose locations are defined by the addressing mode. A "destination" operand is one that is replaced by a result, or is in some other way affected by the instruction. The destination operand is listed first in an addressing mode expression (except in the case of the ST instruction). A "source" operand is a value that is moved or manipulated by the instruction, but is not altered. The source operand is listed second in an addressing mode expression.

The addressing modes described below define how the locations of the source and destination operands are determined. They are divided into three categories:

ONE-ADDRESS GENERAL

These addressing modes are available in all instructions that use general addressing modes. In the class of instructions supporting two addresses (see Table 4-2), these modes specify implicitly that the A register (Accumulator) is the destination, and that the single addressing expression refers to the source operand. (For example, "LD A, [B].W" means to load the Accumulator with the word pointed to by the B register.) In the class of instructions supporting one address (see Table 4-3), the single addressing expression specifies the address of a single operand in memory, which is usually a destination operand. (For example, "INC [B]W" means to increment the word pointed to by the B register.)

TWO-ADDRESS GENERAL These addressing modes are available to the large class of instructions that support two addresses (see Table 4-2). These modes allow memory-to-memory operations using the Direct-Direct mode, immediate-to-memory operations using the Immediate-Direct immediate-to-accumulator mode, and operations using the Immediate mode.

TABLE 4-1. GENERAL ADDRESSING MODES

MODE	SYN ONE-ADDRESS	TAX TWO-ADDRESS	OPERANDS		
B Indirect	[B].B [B].W	A,[B].B A,[B].W	Byte whose address is in B. Word whose address is in B.		
X Indirect	[X].B [X].W	A,[X].B A,[X].W	Byte whose address is in X. Word whose address is in X.		
Direct	address . B address .W	A, address. B A, address. W	Byte at given address. Word at given address.		
Indirect	[pntr].B	A,[pntr].B	Byte whose address is in Basepage pointer word "pntr".		
	[pntr].W	A,[pntr].W	Word whose address is in Basepage pointer word "pntr".		
Indexed	disp[pntr].B	A, disp[pntr]. B	Byte whose address is offset "disp" bytes from contents of Basepage pointer word "pntr".		
	disp[pntr].W	A, disp[pntr].W	Word whose address is offset "disp" bytes from contents of Basepage pointer word "pntr".		
Immediate	_	A, #value	Value given.		
Direct-Direct	_	dest . B , source . B	Destination: Byte at address "dest"; Source: Byte at address "source".		
		dest .W, source . B	Destination: Word at address "dest"; Source: Byte at address "source".		
	_	dest .W, source .W	Destination: Word at address "dest"; Source: Word at address "source".		
	_	dest . B , source .W	Destination: Byte at address "dest"; Source: Word at address "source". (Not Generally Recommended)*		
Immediate-Direct	_	dest . B , #value	Destination: Byte at address "dest"; Source: Value given. **		
	_	dest .W, #value	Destination: Word at address "dest"; Source: Value given.		
*	* This combination of sizes is not recommended except in the IFGT instruction; in all other instructions the result is identical to the "dest.B, source.B" case, which should be used instead.				
**	** The immediate value should always be an 8-bit value; use of a 16-bit value is not recommended.				

TABLE 4-2. INSTRUCTIONS SUPPORTING TWO ADDRESSES

LD			Movement
ADD	ADC	SUBC	Add/Subtract
MULT	DIV	DIVD	Multiply/Divide
DADC	DSUBC		Decimal Add/Subtract
AND	OR	XOR	Logical
IFEQ	IFGT		Relational Tests

TABLE 4-3. INSTRUCTIONS SUPPORTING ONE ADDRESS

IFBIT	#n, mode	RBIT	#n, mode	SBIT	#n, mode	Bit Operations
DECSZ	mode	INC	mode			Memory Inc./Dec.
ST	A, mode	X	A, mode			Store/Exchange A

SPECIALIZED

These modes are limited to relatively small groups of instructions, and thus are not in any sense "general" modes. However, they do add great power and efficiency, especially to data movement instructions. The Specialized modes include the ability to auto-increment or auto-decrement the X or B register in the LD/LDS (Load) and X/XS (Exchange) instructions, and the ability to specify any bit dynamically within 8 Kbytes from a base address in the Bit instructions SBIT, RBIT and IFBIT.

Some instructions use none of the addressing modes discussed in the following pages; these include jumps, subroutine returns, and instructions that operate on the Accumulator or the Carry bit only. Such instructions are referred to as "No-Address" instructions.

The size of an operand may be an 8-bit Byte (mode.B) or 16-bit Word (mode.W), and when using a Two-Address mode the two operands are allowed to differ in size (for example, "ADD 16.W, 21.B"). Word size operands use two bytes of memory and their address must always be even; byte size operands may be stored at even or odd addresses.

NOTE: Depending on the mode of operation selected at Reset (Section 10.1.3), Word accesses may be forbidden outside the on-chip address range 0000—01FF and E000-FFFF. See Section 10.9 for details.

A distinction is made between operands in the first 256 bytes of the addressing range ("Basepage" operands: addresses 0000—00FF) and those in other places. It is generally more efficient to access a Basepage operand, due to the fact that the instruction can use an 8-bit addressing field rather than a 16-bit field. It is also necessary, in the Indirect and Indexed modes, for the pointer variable to be a word, at an even address within the Basepage range.

Note that all the registers in the programming model, and also several important peripheral registers, are memory-mapped within the Basepage range. It is therefore possible, using the Indirect addressing mode, to say "LD A, [K]. B", even though K Register Indirect is not in itself an addressing mode. The remainder of the Basepage space is occupied by on-chip RAM.

4.2 ONE-ADDRESS GENERAL MODES

4.2.1 B Register Indirect

This is the most efficient mode of addressing; nearly all instructions using this addressing mode are only one byte in length. The B register contains the address of the operand.

Example: LD A, [B].W before: A contains 0000 Hex

B contains 0026 Hex

Word at 0026 contains F0F0 Hex

after: A contains F0F0 Hex

B contains 0026 Hex

Word at 0026 contains F0F0 Hex

4.2.2 X Register Indirect

The X register contains the address of the operand. Instructions using this mode are two bytes long in general, although some data movement instructions (LD, X) have optimized forms that use only one byte.

Example: LD A, [X] W before: A contains 0200 Hex

X contains FE0F Hex

Word at FE0F contains C003 Hex

after: A contains C003 Hex

X contains FE0F Hex

Word at FE0F contains C003 Hex

4.2.3 Direct

Instructions using this addressing mode contain an 8-bit or 16-bit address field, which contains the actual address of the operand.

Example: LD A, H'F200.W before: A contains 0000 Hex

Word at F200 contains 0105 Hex

after: A contains 0105 Hex

Word at F200 contains 0105 Hex

4.2.4 Indirect

Instructions using this addressing mode contain an 8-bit address field. This field addresses a pointer word in the Basepage that contains the effective address of the operand. The address of the pointer word must be even.

Example: LD A, [H'1E.W]. B before: A contains 5555 Hex

Word at 001E contains 2CFF Hex Byte at 2CFF contains 78 Hex

after: A contains 0078 Hex

Word at 001E contains 2CFF Hex Byte at 2CFF contains 78 Hex

4.2.5 Indexed

This addressing mode is just like the Indirect addressing mode above except that an additional displacement value is added to obtain the final effective address. Instructions using the Indexed addressing mode contain an 8-bit address field and an 8-bit or 16-bit displacement field. The contents of the Basepage word addressed by the 8-bit address (which must be even) are added to the displacement to obtain the effective address of the operand.

Example: LD A, -2 [H'C4.W].W before: A contains 0000 Hex

Word at 00C4 contains 01C8 Hex Word at 01C6 contains 2344 Hex

after: A contains 2344 Hex

Word at 00C4 contains 01C8 Hex Word at 01C6 contains 2344 Hex

Note that 00C4 is the memory-mapped address of the Stack Pointer. This instruction has the effect of copying the word at the top of the stack into the Accumulator (without altering the Stack Pointer).

4.3 TWO-ADDRESS GENERAL MODES

4.3.1 Immediate

Instructions using the Immediate addressing mode contain an 8 or 16-bit immediate value. This immediate value constitutes the source operand. The Accumulator is the destination operand.

Example: LD A, #H'B9 before: A contains 5555 Hex

after: A contains 00B9 Hex

4.3.2 Direct Memory to Direct Memory

This addressing mode is similar to the Direct addressing mode, except that the destination is another directly addressed memory location rather than the Accumulator. The instruction contains two independent address fields, each being either 8 or 16 bits wide. One field contains the source operand's address and the other field contains the destination operand's address. Both operands may be byte or word in size, although using a 16-bit source operand with an 8-bit destination is not generally a useful combination; the result is usually identical to the case where the source is also 8 bits wide.

Example: LD H'FF40.W, H'40.W before: Word at FF40 contains 0000 Hex

Word at 0040 contains 2826 Hex

after: Word at FF40 contains 2826 Hex

Word at 0040 contains 2826 Hex

4.3.3 Immediate to Direct Memory

The instruction contains an 8- or 16-bit immediate value and an 8- or 16-bit address field. The immediate value is the source operand and the address field contains the address of the destination. Both operands may be byte or word size; the immediate operand's size is dependent only on its contents.

Example: LD H'DD04.W, #H'1111 before: Word at DD04 contains 0000 Hex

after: Word at DD04 contains 1111 Hex

4.4 SPECIALIZED MODES

4.4.1 X Register Indirect with Auto-Increment/Decrement

This addressing mode is available with the two data movement instructions LD and X. It operates identically to the X Register Indirect mode, except that the X register is incremented or decremented after the operand is accessed. The increment or decrement feature is selected by the programmer by stating either [X-] or [X+]. If the operand is a byte in length, the X register is adjusted by one; if it is a word (two bytes) in length, the X register is adjusted by two. This feature facilitates manipulation of blocks of memory in loops.

Example: LD A, [X+], W before: A contains 0000 Hex

X contains FE24 Hex

Word at FE24 contains FCFC Hex

after: A contains FCFC Hex

X contains FE26 Hex

Word at FE24 contains FCFC Hex

LD A, [X-], B before: A contains FFFF Hex

X contains F088 Hex

Byte at F088 contains 11 Hex

after: A contains 0011 Hex

X contains F087 Hex

Byte at F088 contains 11 Hex

4.4.2 B Register Indirect with Auto-Increment/Decrement and Conditional Skip

This addressing mode is available with the data movement instructions LDS and XS. The B register is used to address the operand and is then incremented or decremented (by one for byte operands, and by two for word operands) after the operand has been accessed. The B register is then compared with the K register, and the following instruction is skipped if the B register was incremented and is now greater than K, or the B register was decremented and is now less than K. This addressing mode enables the LDS and XS instructions to be used in loops where blocks of data are being manipulated; the K register is initialized with a limit, and when the B register automatically crosses that limit through successive increments or decrements, a skip out of the loop occurs.

A contains 5555 Hex LDS A, [B+]. B before: Examples:

B contains 00F5 Hex K contains 00F6 Hex

Byte at 00F5 contains 22 Hex

after: A contains 0022 Hex

> B contains 00F6 Hex K contains 00F6 Hex

Byte at 00F5 contains 22 Hex

Next instruction isn't skipped, because

B was incremented and has not exceeded K.

A contains 0000 Hex LDS A, [B+].W before:

> B contains 00F6 Hex K contains 00F6 Hex

Word at 00F6 contains FDDF Hex

A contains FDDF Hex after:

> B contains 00F8 Hex K contains 00F6 Hex

Word at 00F6 contains FDDF Hex

Next instruction is skipped

because B was incremented and

is now greater than K.

LDS A, [B-].W before: A contains 0000 Hex

> B contains FFE8 Hex K contains FFE7 Hex

Word at FFE8 contains 3300 Hex

A contains 3300 Hex after:

> B contains FFE6 Hex K contains FFE7 Hex

Word at FFE8 contains 3300 Hex

Next instruction is skipped

because B was decremented and

is now less than K.

4.4.3 Double Register Indirect (Using the B and X Registers)

This addressing mode is a special case of register indirect addressing, available with the bit instructions RBIT, SBIT and IFBIT. In this mode, register B and (register X)÷8 are added together to form the address of a byte in memory. The specific bit to be modified or tested is then addressed by the least-significant three bits of register X. Therefore, by incrementing only X, simple loops can be written to address through bit arrays of up to 64K bits from any starting memory location held in B. One can also modify B to manipulate even longer bit arrays.

Examples: SBIT X, [B], B before: B contains F000 Hex

X contains 0802 Hex

Byte at F100 contains 0000 Hex

after: B contains F000 Hex X contains 0802 Hex

Byte at F100 contains 0004 Hex

RBIT X, [B]. B before: B contains F000 Hex

X contains 0002 Hex

Byte at F000 contains 0564 Hex

after: B contains F000 Hex

X contains 0002 Hex

Byte at F000 contains 0560 Hex

Chapter 5

DETAILED INSTRUCTION DEFINITIONS

5.1 INTRODUCTION

This chapter defines the instruction set of the HPC family. Included are the mnemonic and binary forms for each instruction, and all information necessary to calculate the exact execution time for any form of the instruction.

5.2 ORGANIZATION AND NOTATIONS

The detailed instruction definitions on the pages that follow are divided into the following sections:

Syntax

This section illustrates how the instruction appears in assembly language. The mnemonic is given, followed by the list of operands required (if any).

Lower-case items appearing here (e.g., "dest", "src") stand for operands that are replaced according to the rules for general addressing modes (see Sections 4.2 and 4.3). Other notations may be used, and are explained with the individual instruction.

Operation

This section gives, in abbreviated form, the operation performed by the instruction.

Description

This section describes the function performed by the instruction in detail. Warnings, restrictions, or extra capabilities are explained here.

Attributes

An instruction has two types of attributes: which general addressing modes it supports, and which data sizes are allowed.

The range of addressing modes supported (Chapter 4) is given as one or more of the following attributes:

Two-Address The instruction supports all general addressing modes, including the

Two-Address modes (Section 4.3).

One-Address The instruction supports all One-Address general addressing modes

(Section 4.2).

No-Address No addressing modes apply to the instruction.

Specialized The instruction supports addressing mode(s) that are not general,

but are specific to that instruction. The modes involved are

explained in the instruction description.

The sizes that the operands may have are defined by a list of the following attributes:

Byte The single operand may be a byte.

Word The single operand may be a word.

Byte → Byte The source and destination operands may both be bytes.

Word → Word The source and destination operands may both be words.

Byte - Word The source operand may be a byte with a word destination.

Word \rightarrow Byte The source operand may be a word with a byte destination.

Carry Bit

This section indicates whether the PSW C bit (Carry) is affected by the instruction.

Encodings

The tables appearing in this section list the binary Opcode value (in hexadecimal) and the resulting length of the instruction in bytes for every available option. An asterisk in a Length column indicates that the instruction is a prefixed form; that is, it starts with an Addressing Directive byte rather than the Opcode byte. Refer to the appropriate addressing mode in Appendix E for full details of encodings.

Timing Information

The tables in this section hold the additional information necessary for calculating the execution times for all forms of the instruction.

The column Min. Exec. Time presents, in units of the CK2 clock, the minimum execution time for each form of the instruction. This time includes all instruction fetches required, and is exact for any system with no Wait states and no Hold states occurring.

When Wait states are inserted in memory accesses, the execution time for any instruction can still be calculated exactly, but the time taken by each Wait state must be added to the execution time. To assist in this, sufficient information is listed to use the following expression:

T	=C+	-(W	*L)	+(V	$V_d * A$)
- exex		•			u -	

where:	C	is the value from the Minimum Execution Time ("Cycles") entry from the Timing Information tables.
	W_{i}	is the number of Wait states applied to instruction fetches.
	L	is the length in bytes of the instruction, taken from the Length entry of the Encodings Table.
	W_d	is the number of Wait states applied to data references. Since this can differ from W_i (EPROM vs. on-chip RAM, for example), it is accounted for separately.
	A	is the number of data accesses performed by the instruction. When a pair of numbers appears (e.g., "3/2"), this means that some of the accesses are Basepage accesses, which can generally be disregarded for purposes of calculating execution times (see Section 10.4), and the second (smaller) number applies instead. The full number of accesses given in the first value do occur (for example, in ROMless modes, they can be seen on the bus), but, since the Basepage accesses are not susceptible to WAIT states, their occurrence is already accounted for in the "C" term of the expression above, regardless of the number of WAIT states generally inserted on data accesses.

Examples

A set of examples is given, showing how the syntax, encodings and timings vary with the different forms of the instruction.

The Symbolic column entry gives an exact form of the instruction in assembly language. For ease of interpretation, some symbols are used, having the following interpretations:

BASEB is a byte in the Basepage addressing range at address 0055 Hex.

BASEW is a word in the Basepage addressing range at address 00AA Hex.

PNTR is a pointer word in the Basepage addressing range at address 0088 Hex.

HIMEMB is a byte in high (i.e., non-Basepage) memory, at address 6677 Hex.

HIMEMW is a word in high (i.e., non-Basepage) memory, at address 3344 Hex.

For branching instructions, the encoding varies with the distance from the branching instruction to the target of the branch. Some label values are assumed, as follows:

HERE is an instruction at address F100 Hex. This is used as the address of the example instruction.

NEAR is an instruction at address F104 Hex.

CLOSE is an instruction at address F130 Hex.

MIDWAY is an instruction at address F050 Hex.

FAR is an instruction at address F808 Hex.

The Addr. Mode column entry gives the addressing mode that the Assembler generates for the instruction example.

The Encoding column entry gives the binary form of the instruction. The individual bytes are listed in hexadecimal, with the first (leftmost) byte listed appearing in memory at the first memory address.

The Illustrative Execution Time entry shows one possible ("typical") execution time for the example instruction. This time is drawn from the following assumptions, which may or may not be valid for any particular end system:

- Instruction fetches contain one Wait state (W_i=1).
- Data accesses to off-chip locations or on-chip ROM contain one Wait state.
- Accesses to on-chip RAM have no Wait states. (This is not an assumption, but is true always.)
- The stack is assumed to be in on-chip RAM, and therefore pushes and pops have no Wait states.
- Data accesses to unknown locations (for example, "[B].B") are assumed to have one Wait state.

[label:] ADC dest, src Syntax:

 $dest \leftarrow dest + src + C$ Operation:

 $C \leftarrow carry$

Description: The source and destination operands are added, and the result is placed in the destination operand. The Carry bit is incorporated into this result: if it is one, the result is greater by one than the sum of the operands; if it is zero, the result is the

exact sum. The Carry bit is then loaded with the carry from the addition (see

Section 2.3).

This form of addition is intended to support multiple-precision arithmetic. For this use, the carry bit is first reset (using the RC instruction), then ADC is used to add the portions of the multiple-precision values, from least-significant to most-

significant.

Two-Address (see Section 5.2). Attributes:

Sizes: byte → byte, word → word, byte → word (see Section 5.2).

Loaded with carry from addition. See Sections 2.3 and 2.4.2. Carry Bit:

	Byte Source		Word Source		
Addressing Mode	Destination	Opcode	Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	C8	1	E8	1
Reg. X indirect	Accumulator (16 bits)	C8	2*	E8	2*
Direct	Accumulator (16 bits)	C8	3, 4*	E8	3, 4*
Indirect	Accumulator (16 bits)	C8	3*	E8	3*
Indexed	Accumulator (16 bits)	C8	4, 5*	E8	4, 5*
Immediate	Accumulator (16 bits)	E8	4*†	E8	5*†
		Byte Dest.		Word Dest.	
Direct-direct	Direct memory	C8	4-6*	E8	4-6*
Immediate-direct	Direct memory	C8	4, 5*	E8	4-6*

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

indicates addressing modes that are generated by the assembler as the appropriate immediate-to-direct forms (byte-word or word-word), using the fact that the accumulator is memory-mapped at address 00C8.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	4	1
Reg. X indirect	7	1
Direct Source in Basepage Source Non-Basepage	8 10	1/0
Indirect	10	2/1
Indexed 8-bit displacement 16-bit displacement	13 15	2/1 2/1
Immediate 8-bit immediate 16-bit immediate	11 13	2/0 2/0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	13	3/0	15	3/2
Source Non-Basepage	15	3/1	17	3
Immediate-Direct				
8-bit immediate	11	2/0	13	2
16-bit immediate	13	2/0	15	2

Symbolic	Addr. Mode	Carry From Bit	Encoding	Illustrative Exec. Time
ADC A,[B].B	Reg. B Indirect	7	C8	6
ADC A,[X].W	Reg. X Indirect	15	8F E8	10
ADC A, 38.W	Direct	15	96 26 E8	11
ADC A,[H'66].B	Indirect	7	AD 66 C8	14
ADC A, 6[PNTR].W	Indexed	15	A2 06 88 E8	18
ADC A, #H'3CF2	Immediate-Direct	15	86 3C F2 C8 E8	18
ADC BASEW, HIMEMB	Direct-Direct	15	84 66 77 AA E8	21
ADC BASEB, #100	Immediate-Direct	7	82 64 55 C8	15

ADD Add

Syntax: [label:] ADD dest, src

Operation: $dest \leftarrow src + dest$

C ← carry

Description: The source and destination operands are added, and the result is placed in the

destination operand. The Carry bit is loaded with the carry from the addition.

Attributes: Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit: Loaded with the carry from addition. See Sections 2.3 and 2.4.2.

		Byte Source		Word Source	
Addressing Mode	Destination	Opcode	Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	D8	1	F8	1
Reg. X indirect	Accumulator (16 bits)	D8	2*	F8	2*
Direct	Accumulator (16 bits)	D8	3, 4*	F8	3, 4*
Indirect	Accumulator (16 bits)	D8	3*	F8	3*
Indexed	Accumulator (16 bits)	D8	4, 5*	F8	4, 5*
Immediate	Accumulator (16 bits)	<u></u> -t	_	B8	3
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	D8	46*	F8	4-6*
Immediate-direct	Direct memory	D8	4, 5*	F8	4-6*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

[†] The Immediate form of the ADD instruction uses a 16-bit immediate field always, and the carry is taken from bit 15. A more compact form for small immediate values is the ADDS instruction, which generates a carry from bit 7.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	4	1
Reg. X indirect	7	1
Direct Source in Basepage Source Non-Basepage	8 10	1/0
Indirect	10	2/1
Indexed 8-bit displacement 16-bit displacement	13 15	2/1 2/1
Immediate (16-bit)	6	0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	13	3/0	15	3/2
Source Non-Basepage	15	3/1	17	3
Immediate-Direct				
8-bit immediate	11	2/0	13	2
16-bit immediate	13	2/0	15	2

ADD Add (Cont)

Symbolic	Addr. Mode	Carry From Bit	Encoding	Illustrative Exec. Time
ADD A,[B].B	Reg. B Indirect	7	D8	6
ADD A,[X].W	Reg. X Indirect	15	8F F8	10
ADD A, 38.W	Direct	15	96 26 F8	11
ADD A,[H'66].B	Indirect	7	AD 66 D8	14
ADD A, 6[PNTR].W	Indexed	15	A2 06 88 F8	18
ADD A, #H'3CF2	Immediate	15	B8 3C F2	9
ADD BASEW, HIMEMB	Direct-Direct	15	84 66 77 AA F8	21
ADD BASEB, #100	Immediate-Direct	7	82 64 55 D8	15

Syntax:

[label:] ADDS A,#value

Operation:

 $A \leftarrow A + \text{value}$

Description: This instruction adds an 8-bit immediate value to the accumulator. The immediate value is zero-extended to 16 bits, and all 16 bits of the accumulator are affected. Unlike the Immediate mode of the ADD instruction, however, the Carry bit indicates the carry from bit 7 rather than bit 15.

Attributes:

No-Address (see Section 5.2).

Size:

byte (see Section 5.2).

Carry Bit:

Loaded with carry from bit 7 (see Section 2.3).

Encodings:

Addressing Mode	Dest	Word Opcode Length	
(None)	Accumulator	98	2

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(None)	4	0

Symbolic	Encoding	Illustrative Exec. Time
ADDS A, #10	98	6

AND Logical AND

Syntax: [label:] AND dest, src

Operation: dest ← dest AND src

Description: The destination operand is logically ANDed with the source operand, and the

result is stored in the destination operand.

Note that if a byte source is ANDed with a word destination (including the accumulator), this has the effect of clearing the high-order byte of the destination.

(This is due to automatic zero-extension, explained in Section 2.4.2.)

Attributes: Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit: Not Affected.

Addressing Mode	Destination	Byte S Opcode	Source Length	Word Opcode	Source Length
Reg. B indirect	Accumulator (16 bits)	D9	1	F9	1
Reg. X indirect	Accumulator (16 bits)	D9	2*	F9	2*
Direct	Accumulator (16 bits)	D9	3, 4*	F9	3, 4*
Indirect	Accumulator (16 bits)	D9	3*	F9	3*
Indexed	Accumulator (16 bits)	D9	4, 5*	F9	4, 5*
Immediate	Accumulator (16 bits)	99	2	В9	3
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	D9	4-6*	F9	4-6*
Immediate-direct	Direct memory	D9	4, 5*	F9	46*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Add	ressing Mode	Min. Exec. Time	Accesses
Reg. B indir	ect	4	1
Reg. X indi	rect	7	1
Direct			
	Source in Basepage	8	1/0
	Source Non-Basepage	10	1
Indirect		10	2/1
Indexed			
	8-bit displacement	12	2/1
	16-bit displacement	14	2/1
Immediate			
	8-bit immediate	4	0
	16-bit immediate	6	0

	Dest. in Basepage		Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	13	3/0	15	3/2
Source Non-Basepage	15	3/1	17	3
Immediate-Direct				
8-bit immediate	11	2/0	13	2
16-bit immediate	13	2/0	15	2

AND Logical AND (Cont)

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
AND A,[B].B	Reg. B Indirect	D9	6
AND A,[X].W	Reg. X Indirect	8F F9	10
AND A, 38.W	Direct	96 26 F9	11
AND A,[H'66].B	Indirect	AD 66 D9	14
AND A, 6[PNTR].W	Indexed	A2 06 88 F9	17
AND A, #H'3CF2	Immediate	B9 3C F2	9
AND BASEW, HIMEMB	Direct-Direct	84 66 77 AA F9	21
AND BASEB, #100	Immediate-Direct	82 64 55 D9	15

Clear Accumulator CLR A

Syntax: [label:] CLR A

Operation: $A \leftarrow 0$

Description: All bits of the accumulator are cleared.

Attributes: No-Address (see Section 5.2).

Size: word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Dest	Wo Opcode	ord Length
(None)	Accumulator	00	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(None)	2	0

Symbolic	Encoding	Illustrative Exec. Time
CLR A	00	3

COMP A Complement Accumulator

Syntax: [label:] COMP A

Operation: $A \leftarrow \overline{A}$

Description: All bits of the accumulator are complemented; i.e., all one bits become zeroes and

all zero bits become ones.

Following this instruction with an INC A instruction has the effect of performing a two's complement (arithmetic negation) on the original contents of the

accumulator.

Attributes: No-Address (see Section 5.2).

Size: word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Dest	Wo Opcode	
(None)	Accumulator	01	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(None)	2	0

Symbolic	Encoding	Illustrative Exec. Time
COMP A	01	3

Syntax:

[label:] DADC dest, src

Operation:

 $dest \leftarrow dest + src + C$ (decimal)

C ← carry from addition

Description: The source and destination operands are added as BCD (Binary Coded Decimal) values, and the BCD result is placed in the destination operand. The Carry bit is incorporated into this result: if it is one, the result is greater by one than the sum of the operands; if it is zero, the result is the exact sum. The Carry bit is then loaded with the carry from the addition (see Section 2.3).

> This form of addition is intended to support multiple-precision arithmetic. For this use, the carry bit is first reset (using the RC instruction), then DADC is used to add the portions of the multiple-precision values, from least-significant to most-significant. Note that this is the only decimal addition instruction in the HPC instruction set. To perform single-precision decimal addition, the Carry bit should always be reset before executing this instruction.

> If the operands do not contain valid BCD values, the result and carry generated by this instruction are undefined.

Attributes:

Two-Address (see Section 5.2).

byte→byte, word→word, byte→word (see Section 5.2).

Loaded with the carry from the decimal addition. See Sections 2.3 and 2.4.2. Carry Bit:

	Dodina	Byte Source		Word Source	
Addressing Mode	essing Mode Destination		Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	C9	1	E9	1
Reg. X indirect	Accumulator (16 bits)	C9	2*	E9	2*
Direct	Accumulator (16 bits)	C9	3, 4*	E9	3, 4*
Indirect	Accumulator (16 bits)	C9	3*	E9	3*
Indexed	Accumulator (16 bits)	C9	4, 5*	E9	4, 5*
Immediate	Accumulator (16 bits)	E9	4*†	E9	5*†
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	C9	4-6*	E9	4-6*
Immediate-direct	Direct memory	C9	4, 5*	E9	46*

- * indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.
- † indicates addressing modes that are generated by the assembler as the appropriate immediate-to-direct forms (byte-word or word-word), using the fact that the accumulator is memory-mapped at address 00C8.

Timing Information:

Add	lressing Mode	Min. Exec. Time	Accesses
Reg. B indi	rect	6	1
Reg. X ind	irect	9	1
Direct			
	Source in Basepage	10	1/0
	Source Non-Basepage	12	1
Indirect		12	2/1
Indexed			
	8-bit displacement	15	2/1
	16-bit displacement	17	2/1
Immediate			
	8-bit immediate	13	2/0
	16-bit immediate	15	2/0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	15	3/0	17	3/2
Source Non-Basepage	17	3/1	19	3
Immediate-Direct				
8-bit immediate	13	2/0	15	2
16-bit immediate	15	2/0	17	2

DADC Decimal Add with Carry (Cont)

Symbolic	Addr. Mode	Carry From Bit	Encoding	Illustrative Exec. Time
DADC A,[B].B	Reg. B Indirect	7	C9	8
DADC A,[X].W	Reg. X Indirect	15	8F E9	12
DADC A, 38.W	Direct	15	96 26 E9	13
DADC A,[H'66].B	Indirect	7	AD 66 C9	16
DADC A, 6[PNTR].W	Indexed	15	A2 06 88 E9	20
DADC A, #H'3CF2	Immediate-Direct	15	86 3C F2 C8 E9	20
DADC BASEW, HIMEMB	Direct-Direct	15	84 66 77 AA E9	23
DADC BASEB, #100	Immediate-Direct	7	82 64 55 C9	17

Syntax:

[label:] DEC A

Operation:

 $\mathbf{A} \leftarrow \mathbf{A} - \mathbf{1}$

Description: The contents of the accumulator are decremented by one.

Attributes: No-Address (see Section 5.2).

word (see Section 5.2).

Carry Bit:

Not Affected.

Encodings

Addressing Mode	Dest	Wo Opcode	ord Length
(None)	Accumulator	05	1

Timing

Information:

:	Addressing Mode	Min. Exec. Time	Accesses
	(None)	2	0

Symbolic	Encoding	Illustrative Exec. Time
DEC A	0.5	3

DECSZ Decrement and Skip if Zero

Syntax: [label:] DECSZ dest

Operation: $dest \leftarrow dest - 1$

if dest = 0 then skip next instruction

Description: The destination operand is decremented by one. If the result is zero, the following

instruction is skipped (not executed).

Attributes: One-Address (see Section 5.2).

Sizes: byte, word (see Section 5.2).

Carry Bit: Not Affected.

Addressing Mode	Destination	В	rte	Word	
Addressing Wode	Destination	Opcode	Length	Opcode	Length
Reg. B indirect	Memory	8A	3*†	AA	3*†
Reg. X indirect	Memory	8A	2*	AA	2*
Direct	Memory				
Basepage		8A	2	AA	2
Non-Basepage		8A	4*	AA	4*
Indirect	Memory	8A	3*	AA	3*
Indexed	Memory	8A	4, 5*	AA	4, 5*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

indicates that this form (B-Indirect) does not exist as a separate opcode. The assembler substitutes the Indirect form, using the fact that the B register is memory-mapped at address OOCC Hex.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	12 or 16+*	3/2
Reg. X indirect	9 or 13+*	2
Direct Basepage Non-Basepage	8 or 12+* 12 or 16+*	2/0 2
Indirect	12 or 16+*	3/2
Indexed 8-bit displacement 16-bit displacement	15 or 19+* 17 or 21+*	3/2 3/2

* The first value is the minimum execution time if no skip occurs. The second value is the minimum execution time when the instruction skips; it must be increased by one for each Wait state imposed on instruction fetches. See Sections 5.2 and 10.4.

Symbolic	Conditions	Encoding	Illustrative Exec. Time
DECSZ [B].B	Reg. B Indirect; No Skip	AD CC 8A	14
DECSZ [X].W	Reg. X Indirect; Skipping	8F AA	18
DECSZ 38.W	Direct; No Skip	96 26 AA	10
DECSZ [H'66].B	Indirect; Skipping	AD 66 8A	22
DECSZ 6[PNTR].W	Indexed; No Skip	A2 06 88 AA	21

DIV Divide

Syntax:

[label:] DIV dest, src

Operation:

dest ← dest ÷ src

X ← remainder $C \leftarrow 0$, $K \leftarrow 0$

Description: The destination operand is divided by the source operand, and the result is placed in the destination operand. The remainder is placed in register X. The K register and the Carry bit are both cleared by this instruction. Both operands are interpreted as unsigned values.

> If division by zero is attempted, the destination receives all ones, the X register receives the original contents of the destination operand, the divisor remains unchanged, the Carry flag is reset, and the K register is cleared.

> Note that this instruction recognizes a dividend that is only 8 or 16 bits in length. As such, it is useful for dividing values in place in memory. See also the instruction DIVD, which divides a 32-bit value by a byte or a word.

Attributes:

Two-Address (see Section 5.2).

byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit:

Cleared always.

		Byte S	Byte Source		Word Source	
Addressing Mode	Destination	Opcode	Length	Opcode	Length	
Reg. B indirect	Accumulator (16 bits)	DF	1	FF	1	
Reg. X indirect	Accumulator (16 bits)	DF	2*	FF	2*	
Direct	Accumulator (16 bits)	DF	3, 4*	FF	3, 4*	
Indirect	Accumulator (16 bits)	DF	3*	FF	3*	
Indexed	Accumulator (16 bits)	DF	4, 5*	FF	4, 5*	
Immediate	Accumulator (16 bits)	9F	2	BF	3	
		Byte	Dest.	Word	Dest.	
Direct-direct	Direct memory	DF	4-6*	FF	4-6*	
Immediate-direct	Direct memory	DF	4, 5*	FF	46*	

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	58	1
Reg. X indirect	61	1
Direct		
Source in Basepage	62	1/0
Source Non-Basepage	64	1
Indirect	64	2/1
Indexed		
8-bit displacement	67	2/1
16-bit displacement	69	2/1
Immediate		
8-bit immediate	58	0
16-bit immediate	60	0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	68	4/0*	70	4/1*
Source Non-Basepage	70	4/3*	72	4*
Immediate-Direct				
8-bit immediate	66	3/0*	68	3*
16-bit immediate	68	3/0*	70	3*

^{*} The dest operand is fetched twice; hence the extra access.

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
DIV A,[B].B	Reg. B Indirect	DF	60
DIV A,[X].W	Reg. X Indirect	8F FF	64
DIV A, 38.W	Direct	96 26 FF	65
DIV A,[H66].B	Indirect	AD 66 DF	68
DIV A, 6[PNTR].W	Indexed	A2 06 88 FF	72
DIV A, #H'3CF2	Immediate	BF 3C F2	63
DIV BASEW, HIMEMB	Direct-Direct	84 66 77 AA FF	76
DIV BASEB, #100	Immediate-Direct	82 64 55 DF	70

DIVD Divide Double Word

[label:] DIVD dest, src Syntax:

 $dest \leftarrow (X, dest) \div src$ Operation:

X - remainder

 $C \leftarrow \text{overflow flag, } K \leftarrow 0$

Description: The 32-bit dividend value consisting of the destination operand (low-order 16 bits) and the X register (high-order 16 bits) is divided by the source operand, and the quotient is placed in the destination operand. The remainder is placed in register X. The K register and the Carry bit are both cleared by the successful completion of this instruction. Both operands are interpreted as unsigned values, and are internally zero-extended to 16-bit values before use. The X register must already contain the top 16 bits of the dividend value before executing this instruction.

> If division by zero is attempted, or if the resulting quotient would be more than 16 bits long (X \geq src), then the Carry bit is set and the X register, the destination operand and the source operand are unchanged. The K register, however, receives an undefined value in this case.

> See also the instruction DIV, which performs an 8 ÷ 8 or 16 ÷ 16 division, and requires no setup.

Two-Address (see Section 5.2). Attributes:

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Cleared if the instruction completes normally; set to one if division by zero is Carry Bit:

attempted or if the result cannot be held in 16 bits.

Addressing Mode	Destination	Byte S Opcode	Source Length	Word Opcode	Source Length
Reg. B indirect	Accumulator (16 bits)	CF	1	EF	1
Reg. X indirect	Accumulator (16 bits)	CF	2*	EF	2*
Direct	Accumulator (16 bits)	CF	3, 4*	EF	3, 4*
Indirect	Accumulator (16 bits)	CF	3*	EF	3*
Indexed	Accumulator (16 bits)	CF	4, 5*	EF	4, 5*
Immediate	Accumulator (16 bits)	CF	4*†	EF	5*†
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	CF	4-6*	EF	46*
Immediate-direct	Direct memory	CF	4, 5*	EF	4-6*

- * indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.
- indicates addressing modes that are generated by the assembler as the appropriate immediate-to-direct forms (byte-word or word-word), using the fact that the accumulator is memory-mapped at address 00C8.

Timing Information:

Addressing Mode		Min. Exec. Time	Accesses	
Reg. B indi	rect	58 or 8†	1	
Reg. X ind	irect	61 or 11†	1	
Direct				
	Source in Basepage	62 or 12†	1/0	
	Source Non-Basepage	64 or 14†	1	
Indirect		64 or 14†	2/1	
Indexed				
•	8-bit displacement	67 or 17†	2/1	
	16-bit displacement	69 or 19†	2/1	
Immediate				
	8-bit immediate	66 or 16†	3/0*	
	16-bit immediate	68 or 18 †	3/0*	

	Dest. in B	asepage	Dest. Non-Basepage		
Addressing Mode	Min. Exec. Time Accesses		Min. Exec. Time	Accesses	
Direct-Direct					
Source in Basepage	68 or 18†	4/0*	70 or 20†	4/3*	
Source Non-Basepage	70 or 20†	4/1*	72 or 22†	4*	
Immediate-Direct					
8-bit immediate	66 or 16†	3/0*	68 or 18†	3*	
16-bit immediate	68 or 18†	3/0*	70 or 20†	3*	

^{*} The dest operand is fetched twice; hence the extra access.

[†] The second time listed is used if an overflow is detected or if division by zero is attempted.

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
DIVD A,[B].B.	Reg. B Indirect	CF	60
DIVD A,[X].W	Reg. X Indirect	8F EF	64
DIVD A, 38.W	Direct	96 26 EF	66
DIVD A,[H'66].B	Indirect	AD 66 CF	68
DIVD A, 6[PNTR].W	Indexed	A2 06 88 EF	72
DIVD A, #H'3CF2	Immediate	86 3C F2 C8 EF	73
DIVD BASEW, HIMEMB	Direct-Direct	84 66 77 AA EF	76
DIVD BASEB, #100	Immediate-Direct	82 64 55 CF	70

DSUBC Decimal Subtract with Carry

Syntax:

[label:] DSUBC dest, src

Operation:

 $dest \leftarrow dest - src - \overline{C}$ (decimal)

 $C \leftarrow carry$

Description: The source and destination operands are subtracted as BCD (Binary Coded Decimal) values, and the BCD result is placed in the destination operand. The Carry bit is incorporated into this result: if it is zero, the result is less by one than the difference of the operands; if it is one, the result is the exact difference. The Carry bit is then loaded with the carry from the subtraction (see Section 2.3).

> This form of subtraction is intended to support multiple-precision arithmetic. For this use, the carry bit is first set (using the SC instruction), then DSUBC is used to subtract the portions of the multiple-precision values, from least-significant to most-significant. Note that this is the only decimal subtraction instruction in the HPC instruction set. To perform single-precision decimal subtraction, the Carry bit should always be set before executing this instruction.

> If the operands do not contain valid BCD values, the result and carry generated by this instruction are undefined.

Attributes:

Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit:

Loaded with the carry from the subtraction (see Sections 2.3 and 2.4.2):

indicates that a borrow condition occurred.

indicates that no borrow condition occurred.

Addressing Mode	ing Mode Destination Byte Source Opcode Length		Word Source Opcode Length		
Reg. B indirect	Accumulator (16 bits)	CA	1	EA	1
Reg. X indirect	Accumulator (16 bits)	CA	2*	EA	2*
Direct	Accumulator (16 bits)	CA	3, 4*	EA	3, 4*
Indirect	Accumulator (16 bits)	CA	3*	EA	3*
Indexed	Accumulator (16 bits)	CA	4, 5*	EA	4, 5*
Immediate	Accumulator (16 bits)	EA	4*†	EA	5*†
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	CA	4-6*	EA	4-6*
Immediate-direct	Direct memory	CA	4, 5*	EA	4-6*

- * indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.
- † indicates addressing modes that are generated by the assembler as the appropriate immediate-to-direct forms (byte-word or word-word), using the fact that the accumulator is memory-mapped at address 00C8.

Timing Information:

Addressing Mode		Min. Exec. Time	Accesses	
Reg. B indi	rect	5	1	
Reg. X indi	rect	8	1	
Direct				
	Source in Basepage	9	1/0	
	Source Non-Basepage	11	1	
Indirect		11	2/1	
Indexed				
	8-bit displacement	14	2/1	
	16-bit displacement	16	2/1	
Immediate				
	8-bit immediate	14	3/0*	
	16-bit immediate	16	3/0*	

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	16	4/0*	18	4/3*
Source Non-Basepage	18	4/1*	20	4*
Immediate-Direct				
8-bit immediate	14	3/0*	16	3*
16-bit immediate	16	3/0*	18	3*

The dest operand is fetched twice; hence the extra access.

Symbolic	Addr. Mode	Carry From Bit	Encoding	Illustrative Exec. Time
DSUBC A,[B].B	Reg. B Indirect	7	CA	7
DSUBC A,[X].W	Reg. X Indirect	15	8F EA	11
DSUBC A, 38.W	Direct	15	96 26 EA	12
DSUBC A,[H'66].B	Indirect	7	AD 66 CA	15
DSUBC A, 6[PNTR].W	Indexed	15	A2 06 88 EA	19
DSUBC A, #H'3CF2	Immediate-Direct	15	86 3C F2 C8 EA	21
DSUBC BASEW, HIMEMB	Direct-Direct	15	84 66 77 AA EA	24
DSUBC BASEB, #100	Immediate-Direct	7	82 64 55 CA	18

IFBIT Test Bit

Syntax:

[label:] IFBIT n, src

X, [B].B [label:] IFBIT

Operation:

If bit n of src = 0, skip next instruction.

Description: This instruction tests a bit in memory. If the bit is set, the following instruction is executed. If it is reset, the following instruction is skipped (not executed), and the instruction following the skipped instruction is executed.

> In the first form listed above, "n" is a constant in the range 0-7. The bit at this position in the destination byte operand is accessed.

> The second form represents a specialized form of addressing allowed in the bit instructions IFBIT, RBIT and SBIT (see Section 4.4.3). Register B points to a byte in memory. Register X contains a bit offset in the range of 0-65535, which is applied relative to bit 0 of this byte, thus allowing any bit within 8K bytes of that byte to be accessed by this instruction. More formally, the byte address and bit position for the accessed bit are calculated as follows:

 $Address = B + (X \div 8)$

Bit No. = X modulo 8

where "B" and "X" represent the contents of the B and X registers.

Attributes:

One-Address (see Section 5.2).

byte (see Section 5.2). Size:

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Destination	Byte		
Addressing wode	Descination	Opcode	Length	
Reg. B indirect	Memory Bit	10-17	1	
Reg. X indirect	Memory Bit	10-17	2*	
Direct	Memory Bit	10-17	3, 4*	
Indirect	Memory Bit	10-17	3*	
Indexed	Memory Bit	10-17	4, 5*	
Reg. B/X Indirect	Memory Bit	3A	1	

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	6 or 10+*	1
Reg. X indirect	9 or 13+*	1
Direct Basepage Non-Basepage	10 or 14+* 12 or 16+*	1/0 1
Indirect	12 or 16+*	2/1
Indexed 8-bit displacement 16-bit displacement	15 or 19+* 17 or 21+*	2/1 2/1
Reg. B/X Indirect	13 or 17+*	1

^{*} The first value is the minimum execution time if no skip occurs. The second value is the minimum execution time when the instruction skips; it must be increased by one for each Wait state imposed on instruction fetches. See Sections 5.2 and 10.4.

IFBIT Test Bit (Cont)

Symbolic	Addr. Mode, Condition	Encoding	Illustrative Exec. Time
IFBIT 7,[B].B	Reg. B Indirect, No Skip	17	8
IFBIT 3,[X].W	Reg. X Indirect, Skipping	8F 13	17
IFBIT 2, 38.W	Direct, No Skip	96 26 12	13
IFBIT 5,[H'66].B	Indirect, No Skip	AD 66 15	16
IFBIT 0,6[PNTR].W	Indexed, Skipping	A2 06 88 10	25
IFBIT X,[B].B	Reg. B/X Indirect, No Skip	3A	15

Syntax:

[label:] IFC

Operation:

Execute next instruction only if C = 1.

Description: If the Carry bit is a one, the instruction following IFC is executed. Otherwise, the

following instruction is skipped (not executed).

Attributes: No-Address (see Section 5.2).

Size:

Not Applicable.

Carry Bit:

Not Affected.

Encodings:

Addressing Mode	Dest	Word Opcode Length	
(None)	(None)	07	1

Timing Information:

Condition	Min. Exec. Time	Accesses
Not Skipping	3 7+*	0
Skipping	/+*	0

This value must be increased by one for each Wait state imposed on instruction fetches. See Sections 5.2 and 10.4.

Symbolic	Condition	Encoding	Illustrative Exec. Time
IFC	Skipping	07	9
IFC	Not Skipping	07	4

IFEQ If Equal

Syntax: [label:] IFEQ src1, src2

Operation: If src1 \neq src2 then skip next instruction.

Description: The two operands are compared. If they are not equal, the following instruction

is skipped (not executed).

Attributes: Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Destination	Byte Source		Word Source	
Addressing Mode	Destination	Opcode	Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	DC	1	FC	1
Reg. X indirect	Accumulator (16 bits)	DC	2*	FC	2*
Direct	Accumulator (16 bits)	DC	3, 4*	FC	3, 4*
Indirect	Accumulator (16 bits)	DC	3*	FC	3*
Indexed	Accumulator (16 bits)	DC	4, 5*	FC	4, 5*
Immediate	Accumulator (16 bits)	9C	2	BC	3
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	DC	46*	FC	46*
Immediate-direct	Direct memory	DC	4, 5*	FC	4-6*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	6 or 10+*	1
Reg. X indirect	9 or 13+*	1
Direct		
Source in Basepage	10 or 14+*	1/0
Source Non-Basepage	12 or 16+*	1
Indirect	12 or 16+*	2/1
Indexed		
8-bit displacement	14 or 18+*	2/1
16-bit displacement	16 or 20+*	2/1
Immediate		
8-bit immediate	6 or 10+*	0
16-bit immediate	8 or 12+*	0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	14 or 18+*	2/0	16 or 20+*	2/1
Source Non-Basepage	16 or 20+*	2/1	18 or 22+*	2
Immediate-Direct				
8-bit immediate	12 or 16+*	1/0	14 or 18+*	1
16-bit immediate	14 or 18+*	1/0	16 or 20+*	1

The first value is the minimum execution time if no skip occurs. The second value is the minimum exection time when the instruction skips; it must be increased by one for each wait state imposed on instruction fetches. See Sections 5.2 and 10.4.

$IFEQ \quad \hbox{ if Equal (Cont)} \\$

Symbolic Addr. Mode, Condition		Encoding	Illustrative Exec. Time
IFEQ A,[B].B	Reg. B Indirect, No Skip	DC	8
IFEQ A,[X].W	Reg. X Indirect, Skipping	8F FC	17
IFEQ A, 38.W	Direct, No Skip	96 26 FC	14
IFEQ A,[H'66].B	Indirect, No Skip	AD 66 DC	16
IFEQ A, 6[PNTR].W	Indexed, Skipping	A2 06 88 FC	24
IFEQ A, #H'3CF2	Immediate, No Skip	BC 3C F2	11
IFEQ BASEW, HIMEMB	Direct-Direct, No Skip	84 66 77 AA FC	22
IFEQ BASEB, #100	Immediate-Direct, Skipping	82 64 55 DC	21

Syntax: [label:] IFGT src1, src2

Operation: If $src1 \le src2$, then skip next instruction.

Description: The two operands are compared as unsigned numbers. If the src1 operand is not

greater than the src2 operand, the following instruction is skipped (not executed).

Attributes: Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word, word→byte (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

A 22	Destination	Byte Source		Word Source	
Addressing Mode	Destination	Opcode	Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	DD	1	FD	1
Reg. X indirect	Accumulator (16 bits)	DD	2*	FD	2*
Direct	Accumulator (16 bits)	DD	3, 4*	FD	3, 4*
Indirect	Accumulator (16 bits)	DD	3*	FD	3*
Indexed	Accumulator (16 bits)	DD	4, 5*	FD	4, 5*
Immediate	Accumulator (16 bits)	9D	2	BD	3
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	DD	4-6*	FD	4-6*
Immediate-direct	Direct memory	DD	4, 5*	FD	4-6*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

IFGT If Greater Than (Cont)

Timing Information:

Add	lressing Mode	Min. Exec. Time	Accesses
Reg. B indi	rect	5 or 9+*	1
Reg. X ind	irect	8 or 12+*	1
Direct			
	Source in Basepage	9 or 13+*	1/0
	Source Non-Basepage	11 or 15+*	1
Indirect		11 or 15+*	2/1
Indexed			
	8-bit displacement	13 or 17+*	2/1
	16-bit displacement	15 or 19+*	2/1
Immediate			
	8-bit immediate	5 or 9+*	0
	16-bit immediate	7 or 11+*	0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	15 or 19+*	3/0†	17 or 21+*	3/2†
Source Non-Basepage	17 or 21+*	3/1†	19 or 23+*	3†
Immediate-Direct				
8-bit immediate	13 or 17+*	2/0†	15 or 19+*	2†
16-bit immediate	15 or 19+*	2/0†	17 or 21+*	2†

- * The first value is the minimum execution time if no skip occurs. The second value is the minimum execution time when the instruction skips; it must be increased by one for each Wait state imposed on instruction fetches. See Sections 5.2 and 10.4.
- † The src1 operand is fetched twice; hence the extra access.

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
IFGT A,[B].B	Reg. B Indirect, No Skip	DD	7
IFGT A,[X].W	Reg. X Indirect, Skipping	8F FD	16
IFGT A, 38.W	Direct, No Skip	96 26 FD	13
IFGT A,[H'66].B	Indirect, No Skip	AD 66 DD	15
IFGT A, 6[PNTR].W	Indexed, Skipping	A2 06 88 FD	23
IFGT A, #H'3CF2	Immediate, No Skip	BD 3C F2	10
IFGT BASEW, HIMEMB	Direct-Direct, No Skip	84 66 77 AA FD	23
IFGT BASEB, #100	Immediate-Direct, Skipping	82 64 55 DD	22

IFNC If Not Carry

Syntax:

[label:] IFNC

Operation:

Execute next instruction only if C = 0.

Description: If the Carry bit is a zero, the instruction following IFNC is executed. Otherwise,

the following instruction is skipped (not executed).

Attributes: No-Address (see Section 5.2).

Size:

Not Applicable.

Carry Bit:

Not Affected.

Encodings:

Addressing Mode	Dest		ord Length
(None)	(None)	06	1

Timing Information:

Condition	Min. Exec. Time	Accesses
Not Skipping	3	0
Skipping	7+*	0

This value must be increased by one for each Wait state imposed on instruction fetches. See Sections 5.2 and 10.4.

Symbolic	Condition	Encoding	Illustrative Exec. Time
IFNC	Skipping	06	9
IFNC	Not Skipping	06	4

Syntax:

[label:] INC dest

Operation:

 $dest \leftarrow dest + 1$

Description: The destination operand is incremented by one.

Attributes:

One-Address (see Section 5.2).

Sizes: byte, word (see Section 5.2).

Carry Bit:

Not Affected.

Encodings:

Addmarain - Norde	Destination	Byte		Word	
Addressing Mode	Destination	Opcode	Length	Opcode	Length
Reg. B indirect	Memory	89	3*†	A9	3*†
Reg. X indirect	Memory	89	2	A9	2
Direct	Memory				
Basepage		89	2	A9	2
Non-Basepage		89	4*	A9	4*
Indirect	Memory	89	3*	A9	3*
Indexed	Memory	89	4, 5*	A9	4, 5*

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

[†] indicates that this form (B-Indirect) does not exist as a separate opcode. The assembler substitutes the Indirect form, using the fact that the B register is memory-mapped at address 00CC Hex.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	11	3/2
Reg. X indirect	8	2
Direct		
Basepage	7	2/0
Non-Basepage	11	2
Indirect	11	3/2
Indexed		
8-bit displacement	14	3/2
16-bit displacement	16	3/2

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
INC A,[B].B	Reg. B Indirect	AD CC 89	16
INC A,[X].W	Reg. X Indirect	8F A9	12
INC A, 38.W	Direct	96 26 A9	10
INC A,[H'66].B	Indirect	AD 66 89	16
INC A, 6[PNTR].W	Indexed	A2 06 88 A9	20

Increment Accumulator INC A

Syntax: [label:] INC A

Operation: $A \leftarrow A + 1$

Description: The contents of the accumulator are incremented by one.

Attributes: No-Address (see Section 5.2).

Size: word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Dest		ord Length
(None)	Accumulator	04	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(None)	2	0

Symbolic	Encoding	Illustrative Exec. Time
INC A	04	3

\mathbf{IIID} Jump Indirect (Byte)

Syntax:

[label:] JID

followed by

[label:] .PT dest0, dest1, dest2, . . .

Operation:

PC ← PC + Jump Table entry selected by Accumulator contents

Description: This one-byte instruction performs a multi-way branch (similar to the "switch" statement in the C language). It is followed by a table of one-byte entries (a Jump Table) generated by the following ".PT" directive(s). (The .PT directive may be repeated as often as necessary to list all table entries.) Each table entry points to a separate destination for the jump. The 16-bit contents of the accumulator determine which Jump Table entry is selected: zero selects the first entry, one selects the second, and so on.

> Each table entry contains an unsigned "self-relative" displacement; that is, the entry contains the difference between its own address and that of the destination instruction. Because the entries are unsigned, the JID instruction is only allowed to jump forward, and the destinations are allowed to be placed up to 255 bytes ahead of their table entries. See also the JIDW instruction, which allows jumps anywhere within the 64K-byte addressing range of the HPC.

Refer to the HPC Assembler/Linker/Librarian User's Manual, NSC Publication Number 424410836-001 for information on the ".PT" assembler directive.

Attributes:

No-Address (see Section 5.2).

Size: None.

Carry Bit:

Not Affected.

Encodings:

Opcode	Length
CC	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Table Relative	7	1

Examples:

Sy	mbolic	Encoding	Illustrative Exec. Time
HERE:	JID	cc	9
	.PT NEAR	03	

This example assumes that the accumulator contains zero, and shows only the . PT table entry that is relevant to this case.

MOIL Jump Indirect (Word)

Syntax:

.ODD followed by

label:

JIDW

and

label:

.PTW dest0, dest1, dest2, . . .

Operation:

PC ← PC + Jump Table entry selected by Accumulator contents

Description: This one-byte instruction performs a multi-way branch (similar to the "switch" statement in the C language). It is followed by a table of 16-bit entries (a Jump Table) generated by the following ". PTW" directive(s). (The . PTW directive may be repeated as often as necessary to list all table entries.) Each table entry points to a separate destination for the jump.

> The 16-bit contents of the accumulator determine which Jump Table entry is selected, but in a manner slightly different from the JID instruction: the contents must be even, and successive even values select the table entries. Zero selects the first entry, two selects the second, four selects the third, and so on. Proper accumulator contents can be generated simply by loading the accumulator with the desired index value (as per the JID instruction), and then shifting the contents of the accumulator left by one bit position (using "SHL A") before executing JIDW.

> Each table entry contains a signed (two's complement) "self-relative" displacement; that is, the entry contains the difference between its own address and that of the destination instruction.

- NOTES: 1. Because the table entries are words, they must be placed only at even addresses. Because of this, the one-byte JIDW opcode must be located at an odd address. This can be ensured by always preceding the JIDW instruction with a ".ODD" assembler directive. This directive generates a one-byte NOP instruction, if necessary, to place the following instruction (JIDW) at an odd address.
 - 2. The JIDW instruction cannot be used in 8-bit external memory. This is because the Jump Table entry is fetched in a single 16-bit bus transfer for reasons of efficiency. The same effect, however, can be obtained in 8-bit memory by using the JID instruction to jump into a set of JMP and/or JMPL instructions.
 - 3. Refer to the HPC Assembler/Linker/Librarian User's Manual, NSC Publication Number 424410836-001 for information on the ".PTW" assembler directive.

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit: Not Affected.

Encodings:

Opcode	Length
EC	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Table Relative	7	1

Symbolic	Encoding	Illustrative Exec. Time
SHL A HERE: .ODD JIDW .PTW CLOSE, MIDWAY, FAR	E7 40 (NOP at even address) EC 2F 00 4C FF 02 07	4 5 9

JMP Jump Relative

Syntax: [label:] JMP dest

Operation: PC ← PC + displacement to dest

Description: This two-byte instruction performs a jump to an instruction that is within the

range -255 to +257 bytes from the first byte of the JMP instruction. (The assembler will by default substitute the JP opcode if the destination instruction is close enough, or the JMPL opcode if the destination instruction is outside this

range.)

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit: Not Affected.

Encodings:

Direction	Opcode	Length
Forward	94*	2
Backward	95*	2

* The opcode is followed by a one-byte unsigned value, which encodes the distance to the destination instruction. This displacement byte is interpreted according to the direction of the jump:

Forward:

00 forward 2
01 forward 3...

FF forward 257.

Backward:

oo illegal (actually jumps to the second byte of the JMP instruction)

01 backward 0 (jump to self)

02 backward 1...

until

FF backward 255.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
PC Relative	5	0

Symbolic	Encoding	Illustrative Exec. Time
HERE: JMP MIDWAY	95 B0	7

JMPL Jump Relative Long

Syntax:

[label:] JMPL dest

Operation:

 $PC \leftarrow PC + displacement to dest$

Description: This three-byte instruction performs a jump to an instruction that is anywhere within the 16-bit addressing range of the HPC. (The assembler will by default substitute the JP or JMP opcode if the destination instruction is close enough.)

Attributes:

No-Address (see Section 5.2).

Size: None.

Carry Bit:

Not Affected.

Encodings:

Opcode	Length
B4*	3

The opcode is followed by a two-byte signed value (two's complement), which encodes the distance to the destination instruction. This displacement value is interpreted as being relative to the first byte of the instruction following the JSRL instruction in memory. It is appended to the opcode most-significant byte first.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
PC Relative	8	0

Symbolic	Encoding	Illustrative Exec. Time
HERE: JMPL FAR	B4 07 05	11

Syntax:

[label:] JP dest

Operation:

PC ← PC + displacement to dest

Description: This one-byte instruction performs a jump to an instruction that is within the range -31 to +32 bytes from the JP instruction. (The assembler will by default substitute the JMP or JMPL opcode if the destination instruction is outside this range.)

NOTE:

The instruction "JP . + 1", which simply jumps to the next sequential instruction, has the alternate mnemonic "NOP".

Attributes:

No-Address (see Section 5.2).

None. Size:

Carry Bit:

Not Affected.

Encodings:

Opcode (Binary)	Length
01dddddd*	1

The opcode contains a six-bit field (shown by the "d" bits) which encodes the relative displacement to the destination instruction. This value is effectively signed: the most-significant bit of the displacement field indicates the direction of the jump:

1 = backward, by 0 to 31 bytes;

0 =forward, by 1 to 32 bytes,

and the remainder of the field indicates how far the instruction will jump. Note that this is not a complement form of representation, but rather a form of sign-magnitude:

When the displacement field is all zeroes (Opcode = 40 Hex) it represents a jump of one byte forward (NOP). A field of 000001 (41 Hex) jumps two bytes forward, etc., until 011111 (5F Hex), which jumps 32 bytes forward.

For negative displacements, a field of 100000 (60 Hex) represents a backward jump of zero ("JP .", which jumps to itself). A field of 100001 (61 Hex) jumps backward one byte, etc., until 111111 (7F Hex) which jumps backward 31 bytes.

JP

Jump Relative Short (Cont)

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
PC Relative	4	0

Symbolic	Encoding	Illustrative Exec. Time
HERE: JP NEAR	43	5

Syntax:

[label:] JSR dest

Operation:

Word pointed at by SP ← Address of following instruction

 $SP \leftarrow SP + 2$

PC - PC + displacement to dest

Description: This two-byte instruction performs a jump to an instruction that is within the range -1023 to +1025 bytes from the first byte of the JSR instruction. (The assembler will by default substitute the JSRP opcode if "dest" has been declared using the ".SPT" directive, or the JSRL opcode if the destination instruction is

outside this range.)

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit:

Not Affected.

Encodings:

Opcode (Binary)	Length
00110ddd dddddddd *	2

The opcode contains an 11-bit field (shown by the "d" bits) which encodes the relative displacement to the destination instruction. This value is effectively signed: the mostsignificant bit of the displacement field (in the first byte of the opcode) indicates the direction of the jump:

0 = Forward:

forward 3 000 00000000 000 00000001 forward 4... until forward 1025. 011 111111111

1 = Backward:

illegal (actually jumps to the third byte of the JSR 100 00000000 instruction) illegal (actually jumps to the second byte of the JSR 100 00000001 instruction)

100 00000010

backward 0 (jump to self)

100 00000011

backward 1...

until

111 111111111

backward 1023.

Note that this is not a complement representation, but rather a form of sign-magnitude.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
PC Relative	8	1

Symbolic	Encoding	Illustrative Exec. Time
HERE: JSR MIDWAY	34 B0	11

Syntax:

[label:] JSR dest

Operation:

Word pointed at by SP ← Address of following instruction

 $SP \leftarrow SP + 2$

PC ← PC + displacement to dest

Description: This two-byte instruction performs a jump to an instruction that is within the range -1023 to +1025 bytes from the first byte of the JSR instruction. (The assembler will by default substitute the JSRP opcode if "dest" has been declared using the ".SPT" directive, or the JSRL opcode if the destination instruction is

outside this range.)

Attributes:

No-Address (see Section 5.2).

None.

Carry Bit:

Not Affected.

Encodings:

Opcode (Binary)	Length
00110ddd dddddddd *	2

The opcode contains an 11-bit field (shown by the "d" bits) which encodes the relative displacement to the destination instruction. This value is effectively signed: the mostsignificant bit of the displacement field (in the first byte of the opcode) indicates the direction of the jump:

0 = Forward:

000 00000000 forward 3 forward 4... 000 00000001 until forward 1025. 011 11111111

1 = Backward:

illegal (actually jumps to the third byte of the JSR 100 000000000 instruction) illegal (actually jumps to the second byte of the JSR 100 00000001 instruction)

JSR Jump to Subroutine (Cont)

100 00000010

backward 0 (jump to self)

100 00000011

backward 1...

until

111 111111111

backward 1023.

Note that this is not a complement representation, but rather a form of sign-magnitude.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
PC Relative	8	1

Symbolic	Encoding	Illustrative Exec. Time
HERE: JSR MIDWAY	34 B0	11

Syntax:

[label:] JSRL dest

Operation:

Word pointed at by SP ← Address of following instruction

 $SP \leftarrow SP + 2$

PC ← PC + displacement to dest

Description: This three-byte instruction performs a subroutine jump to an instruction that is anywhere within the 16-bit addressing range of the HPC. (The assembler will by default substitute the JSRP opcode if "dest" has been declared with the ". SPT" directive (see the HPC Assembler/Linker/Librarian User's Manual, NSC Publication Number 424410836-001} or the JSR opcode if the destination

instruction is close enough.)

Attributes:

No-Address (see Section 5.2).

None. Size:

Carry Bit:

Not Affected.

Encodings:

Opcode	Length
B5*	3

The opcode is followed by a two-byte signed value (two's complement), which encodes the distance to the destination instruction. This displacement value is interpreted as being relative to the first byte of the instruction following the JSRL instruction in memory. It is appended to the opcode byte most-significant byte first.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
PC Relative	10	1

Symbolic	Encoding	Illustrative Exec. Time
HERE: JSRL FAR	B5 07 05	14

(JSRP) Jump to Subroutine from Table

Syntax:

.SPT dest

followed by either

[label:]

JSR dest

IJΕ

[label:]

JSRL dest

Operation:

Word pointed at by SP ← Address of following instruction

 $SP \leftarrow SP + 2$

PC ← Subroutine Table entry selected by instruction

Description: This one-byte instruction performs a subroutine jump to any instruction whose address has been entered in the Subroutine Table (by declaring it using the assembler directive ". SPT dest"). Up to 16 such subroutines may be so declared (see the HPC Assembler/Linker/Librarian User's Manual, NSC Publication Number 424410836-001).

The mnemonic "JSRP" does not actually exist, but is only an internal NSC designation for the opcode. To cause the assembler to use this opcode, declare the subroutine using the ".SPT" directive, then reference it using either the JSR or JSRL mnemonic.

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit: Not Affected.

Encodings:

Opcode (Binary)	Length
0010nnnn* (20—2F Hex)	1

* The opcode contains a four-bit field (shown by the "n" bits), which selects the desired entry of the Subroutine Table that starts at address FFDO (Hex). Each entry in this table is two bytes long and contains the address of the corresponding subroutine. The desired entry is located by multiplying "n" by two and adding the result to the base address of the table. The table can contain up to 16 such entries, spanning up to 32 bytes in memory. To keep the table as small as possible when not all sixteen entries are used, the assembler allocates entries of this table "backward"; that is, entry number 15 (n = 1111) is allocated to the subroutine declared by the first ". SPT" directive.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Table Indirect	11	3*

* The Subroutine Table entry is accessed using two one-byte read accesses.

Examples

Symbolic	Encoding	Illustrative Exec. Time	
.SPT FAR HERE: JSR FAR	2F*	15	

* This assumes that FAR is the first subroutine in the program that is declared using ". SPT".

LD Load

Syntax: [label:] LD dest, src

Operation: dest ← src

Syntax: [label:] LD A, [X+].B

[label:] LD A,[X+].W [label:] LD A,[X-].B [label:] LD A,[X-].W

Operation: $A \leftarrow \text{value pointed to by } X$

 $X \leftarrow X \pm 1 \text{ or } 2$

Syntax: [label:] LD B, #value

[label:] LD X, #value [label:] LD K, #value

[label:] LD BK, #Bvalue, #Kvalue

Operation: $dest(s) \leftarrow value(s)$

Description: The destination operand is loaded from the source operand. If the destination is longer than the source, the source value is zero-extended to fit (see Section 2.4.2).

Note that registers as well as memory locations may be loaded with this

instruction by using their memory-mapped addresses.

In addition to the general two-address addressing modes, the LD instruction has forms allowing use of X register indirect addressing with auto-increment or auto-decrement, immediate forms allowing the B, X or K register to be loaded more efficiently with a value, and an extended immediate form allowing the B and K registers both to be loaded with two immediate values (a frequent operation in programs performing loops).

See also the LDS instruction, which performs a B register indirect access with

auto-increment or auto-decrement and a conditional skip.

Attributes: Two-Address (see Section 5.2).

Specialized addressing extensions: X Register Indirect with auto-increment or

auto-decrement, and certain immediate-to-register forms.

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit: Not Affected.

Encodings

Addressing Mode	Destination	Byte Source Opcode Length		Word Source Opcode Length			
Reg. B indirect	Accumulator	C4	1		E4		1
Reg. X indirect	Accumulator	D4	1		F4		1
Direct Basepage Non-Basepage	Accumulator	88 88	2 4		A8 A8		2 4*
Indirect	Accumulator	88	31	k	A8		3*
Indexed	Accumulator	88	4, :	5*	A8	4	4, 5*
Immediate	Accumulator	90	2		В0		3
		Byte→Byte Word		d→Word	Byte	→Word	
Direct-direct Src, Dest both in Basepage	Direct memory	8C	3	AC	3	AB	4*
Otherwise	Direct memory	8B	5, 6*	AB	5, 6*	AB	5, 6*
Immediate-direct Dest in Basepage	Direct memory	97	3	В7	4	AB	4*
Dest Non-Basepage	Direct memory	8B	5*	AB	6*	AB	5*
		Byte	Source	е	Word Source		
		Opcode	Len	gth	Opcode	L	ength
X-Indirect Auto-Inc.	Accumulator	D0	1		F0	1	
X-Indirect Auto-Dec.	Accumulator	D2	1		F2	-	1
	Reg. B	92		2	B2		3
Immediate-Register	Reg. X	93		2	B3 B1		3
	Reg. K Regs. B and K	91 8D	ı	2 3	A7		5

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

$LD \quad {\tt Load} \, ({\tt Cont})$

Timing Information:

Addressing Mode		Min. Exec. Time	Accesses
Reg. B indi	rect	4	1
Reg. X ind	irect	4	1
Direct			
1	Source in Basepage	6	1/0
	Source Non-Basepage	10	1
Indirect		10	2/1
Indexed			
	8-bit displacement	13	2/1
	16-bit displacement	15	2/1
Immediate			
	8-bit immediate	4	0
	16-bit immediate	6	0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	10	2/0	16	2/1
Source Non-Basepage	16	2/1	18	2
Immediate-Direct				
8-bit immediate	8	1/0	10	1
16-bit immediate	14	1/0	16	1

Addressing Mode	Min. Exec. Time	Accesses	
Reg. X indirect, Auto-Inc.	4	1	
Reg. X indirect, Auto-Dec.	4	1	

	Ву	e	Word		
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses	
Immediate to B	4	0	6	0	
Immediate to X	4	0	6	0	
Immediate to K	4	0	6	0	
Immediate to B and K	6	0	10	0	

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
LD A,[B].B	Reg. B Indirect	C4	6
LD A,[X].W	Reg. X Indirect	F4	6
LD A, 38.W	Direct	A8 26	8
LD A,[H'66].B	Indirect	AD 66 88	14
LD A, 6[PNTR].W	Indexed	A2 06 88 A8	18
LD A, #H'3CF2	Immediate	B0 3C F2	9
LD BASEW, HIMEMB	Direct-Direct	84 66 77 AA AB	22
LD BASEB, #100	Immediate-Direct	97 64 55	11
LD B, #74	Immediate-B	92 4A	6
LD X, #BASEW	Immediate-X	93 AA	6
LD X, BASEW	Direct-Direct	AC AA CE	13
LD BK, #BASEB, #BASEB+31	Immediate-B&K	8D 55 74	9
LD BK, #H'3F24, #H'3F00	Immediate-B&K	A7 3F 24 3F 00	15
LD A, [X+].W	X-Indir. Auto-Inc.	F0	6
LD A, [X-].B	X-Indir. Auto-Dec.	D2	6

LDS Load A, with Auto-Increment/Decrement and Conditional Skip

Syntax: [label:] LDS [B+]B

[B+].W[label:] LDS [label:] LDS [B-].B [label:] LDS [B-].W

Operation:

 $A \leftarrow \text{operand (pointed to by B)}$

 $B \leftarrow B \pm \text{ operand's width (1 or 2)}$

If B incremented/decremented past address in K, skip next instruction.

Description: The contents of the location addressed by the B register are placed in the accumulator. The B register is then auto-incremented or auto-decremented (as selected by either "+" or "-") by the number of bytes in the operand (i.e., 1 for the Byte form, or 2 for the Word form of the instruction). If, after incrementing B, the result is greater than the value in the K register, the following instruction is skipped (not executed). If, after decrementing B, the result is less than the value in the K register, the following instruction is skipped.

> Note that, since the HPC is an unsigned machine, no value is considered greater than FFFF Hex or less than zero; the K register may therefore be set to one of these values to effectively disable skipping. By the same token, however, neither decrementing while K = 0 nor incrementing while K = FFFF will ever cause a skip (for example, to terminate a loop). Loops involving the first or last byte (or word) of memory, then, should use this instruction with care.

Attributes:

Specialized addressing: using Register B (see Section 4.4.2).

Sizes: byte, word (see Section 5.2).

Carry Bit:

Not Affected.

Encodings:

Addressing Mode	Source	Byte Opcode Length		Wo Opcode	ord Length
Auto-Increment	Memory	CO	1	EO	1
Auto-Decrement	Memory	C2	1	E2	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Without Skip	4	1
With Skip	8+*	1

* This value must be increased by one for each Wait state imposed on instruction fetches. See Sections 5.2 and 10.4.

Symbolic	Conditions	Encoding	Illustrative Exec. Time
LDS A,[B+].B	Auto-Increment by 1, No Skip	C0 E2	6 11
LDS A,[B-].W	Auto-Decrement by 2, Skipping	L'Z	11

MULT Multiply

Syntax: [label:] MULT dest, src

Operation: X&dest ← dest × src

 $K \leftarrow 0$; $C \leftarrow 0$

Description: The source and destination operands are read and internally zero-extended (if

necessary) to 16-bit values. They are then multiplied as unsigned numbers, and the 32-bit result is placed in the X register (most-significant 16 bits) and the destination operand (least-significant 8 or 16 bits). The K register and the Carry bit are cleared. Note that bits 8—15 of the result are not delivered when the

destination is one byte in length.

Attributes: Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit: Cleared.

Encodings:

4.3.3	Destination	Byte Source		Word Source	
Addressing Mode		Opcode	Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	DE	1	FE	1
Reg. X indirect	Accumulator (16 bits)	DE	2*	FE	2*
Direct	Accumulator (16 bits)	DE	3, 4*	FE	3, 4*
Indirect	Accumulator (16 bits)	DE	3*	FE	3*
Indexed	Accumulator (16 bits)	DE	4, 5*	FE	4, 5*
Immediate	Accumulator (16 bits)	9E	2	BE	3
		Byte	Dest.	Word	Dest.
Direct-direct	Direct memory	DE	46*	FE	4-6*
Immediate-direct	Direct memory	DE	4, 5*	FE	4-6*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	58	1
Reg. X indirect	61	1
Direct		
Source in Basepage	62	1/0
Source Non-Basepage	64	1
Indirect	64	2/1
Indexed		
8-bit displacement	67	2/1
16-bit displacement	69	2/1
Immediate		
8-bit immediate	58	0
16-bit immediate	60	0

	Dest. in Basepage		Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	68	4/0*	70	4/3*
Source Non-Basepage	70	4/1*	72	4*
Immediate-Direct				
8-bit immediate	66	3/0*	68	3*
16-bit immediate	68	3/0*	70	3*

^{*} The dest operand is fetched twice; hence the extra access.

MULT Multiply (Cont)

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
MULT A,[B].B	Reg. B Indirect	DE	60
MULT A,[X].W	Reg. X Indirect	8F FE	64
MULT A, 38.W	Direct	96 26 FE	65
MULT A,[H66].B	Indirect	AD 66 DE	68
MULT A, 6[PNTR].W	Indexed	A2 06 88 FE	72
MULT A, #H'3CF2	Immediate	BE 3C F2	63
MULT BASEW, HIMEMB	Direct-Direct	84 66 77 AA FE	76
MULT BASEB, #100	Immediate-Direct	82 64 55 DE	70

Syntax: [label:] NOP

Operation: $PC \leftarrow PC + 1$

Description: This one-byte instruction performs a jump to the following instruction; thereby

performing no operation. This instruction is the same as that generated by stating

"JP . + 1".

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit: Not Affected.

Encodings: Opcod

Opcode (Binary)	Length
40	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses	
(None)	4	0	

Symbolic	Encoding	Illustrative Exec. Time
HERE: NOP	40	5

OR Logical Or

Syntax: [label:] OR dest, src

Operation: dest ← dest OR src

Description: This instruction performs a bitwise logical OR operation between the bits of the

destination and source operands, and places the result into the destination. In byte-to-word cases, the upper byte of the destination is unaffected due to

automatic zero-extension of the source (see Section 2.4.2).

Attributes: Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

		Byte Source		Word Source	
Addressing Mode	Destination	Opcode	Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	DA	1	FA	1
Reg. X indirect	Accumulator (16 bits)	DA	2*	FA	2*
Direct	Accumulator (16 bits)	DA	3, 4*	FA	3, 4*
Indirect	Accumulator (16 bits)	DA	3*	FA	3*
Indexed	Accumulator (16 bits)	DA	4, 5*	FA	4, 5*
Immediate	Accumulator (16 bits)	9A	2	BA	3
			Dest.	Word	Dest.
Direct-direct	Direct memory	DA	4-6*	FA	4-6*
Immediate-direct	Direct memory	DA	4, 5*	FA	46*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addı	ressing Mode	Min. Exec. Time	Accesses
Reg. B indir	ect	4	1
Reg. X indir	ect	7	1
	Source in Basepage Source Non-Basepage	8 10	1/0
Indirect		10	2/1
Indexed	8-bit displacement 16-bit displacement	12 14	2/1 2/1
Immediate	8-bit immediate 16-bit immediate	4 6	0

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	13	3/0	15	3/2
Source Non-Basepage	15	3/1	17	3
Immediate-Direct				İ
8-bit immediate	11	2/0	13	2
16-bit immediate	13	2/0	15	2

$OR \quad \text{Logical Or (Cont)}$

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
OR A,[B].B	Reg. B Indirect	DA	6
OR A,[X].W	Reg. X Indirect	8F FA	10
OR A, 38.W	Direct	96 26 FA	11
OR A,[H'66].B	Indirect	AD 66 DA	14
OR A, 6[PNTR].W	Indexed	A2 06 88 FA	17
OR A, #H'3CF2	Immediate	BA 3C F2	9
OR BASEW, HIMEMB	Direct-Direct	84 66 77 AA FA	21
OR BASEB, #100	Immediate-Direct	82 64 55 DA	15

[label:] POP dest

Operation:

 $SP \leftarrow SP - 2$

dest ← word pointed to by SP

Description: The SP register is auto-decremented by 2. The destination operand is then loaded from the memory location addressed by the contents of the SP register.

> The operand is addressed using the Direct addressing mode, and it must be a word within the first 256 bytes of memory (i.e., a Basepage operand). Note that most non-I/O registers are memory-mapped within this range.

Attributes:

Specialized addressing: Basepage Direct

word (see Section 5.2). Size:

Carry Bit:

Not affected, unless PSW register is the destination.

Encodings

Addressing Mode	Dest	Wo Opcode	ord Length
Basepage Direct	Memory	3F	2

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Basepage Direct	8	2/1

Symbolic	Encoding	Illustrative Exec. Time
POP A	3F C8	11
POP 38.W	3F 26	11

PUSH Push onto the Stack

Syntax: [label:] PUSH src

Operation: Word pointed to by SP ← src

 $SP \leftarrow SP + 2$

Description: The contents of the source operand are placed in the memory location addressed

by the contents of the SP register. The SP register is then auto-incremented by 2.

The operand is addressed using the Direct addressing mode, and it must be a word within the first 256 bytes of memory (i.e., a Basepage operand). Note that most

non-I/O registers are memory-mapped within this range.

Attributes: Specialized addressing: Basepage Direct

Size: word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Source	Wo Opcode	ord Length
Basepage Direct	Memory	AF	2

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Basepage Direct	8	2/1

Symbolic	Encoding	Illustrative Exec. Time
PUSH A	AF C8	11
PUSH 38.W	AF 26	11

[label:] RBIT n, dest

[label:] RBIT X, [B].B

Operation:

bit n of dest $\leftarrow 0$

Description: This instruction clears a bit in memory.

In the first form listed above, "n" is a constant in the range 0-7. The bit at this position in the destination byte operand is accessed.

The second form represents a specialized form of addressing allowed in the bit instructions IFBIT, RBIT and SBIT (see Section 4.4.3). Register B points to a byte in memory. Register X contains a bit offset in the range of 0-65535, which is applied relative to bit 0 of this byte, thus allowing any bit within 8 Kbytes of that byte to be accessed by this instruction. More formally, the byte address and bit position for the accessed bit are calculated as follows:

 $Address = B + (X \div 8)$

Bit No. = X modulo 8

where "B" and "X" represent the contents of the B and X registers.

Attributes:

One-Address (see Section 5.2).

byte (see Section 5.2). Size:

Carry Bit:

Not Affected.

Encodings:

	-	Ву	rte
Addressing Mode	Destination	Opcode	Length
Reg. B indirect	Memory Bit	18-1F	1
Reg. X indirect	Memory Bit	18-1F	2*
Direct	Memory Bit	18-1F	3, 4*
Indirect	Memory Bit	18-1F	3*
Indexed	Memory Bit	18-1F	4, 5*
Reg. B/X Indirect	Memory Bit	38	1

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	5	2
Reg. X indirect	8	2
Direct		
Basepage	9	2/0
Non-Basepage	11	2
Indirect	11	3/2
Indexed		
8-bit displacement	14	3/2
16-bit displacement	16	3/2
Reg. B/X Indirect	12	2

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
RBIT 7,[B].B	Reg. B Indirect	1F	8
RBIT 3,[X].W	Reg. X Indirect	8F 1B	12
RBIT 2, 38.W	Direct	96 26 1A	12
RBIT 5,[H'66].B	Indirect	AD 66 1D	16
RBIT 0,6[PNTR].W	Indexed	A2 06 88 18	18
RBIT X,[B].B	Reg. B/X Indirect	38	15

[label:] RC

Operation:

 $C \leftarrow 0$

Description: The Carry bit is cleared.

Attributes:

No-Address (see Section 5.2).

Size:

Not Applicable.

Carry Bit:

Cleared to zero.

Encodings:

Addressing Mode	Dest	Wo Opcode	ord Length
(None)	Carry bit	03	1

Timing

Information:

Addressing Mode	Min. Exec. Time	Accesses
(None)	2	0

Symbolic	Encoding	Illustrative Exec. Time
RC	03	3

RET Return from Subroutine

Syntax: [label:] RET

Operation: $SP \leftarrow SP - 2$

PC ← word pointed to by SP

Description: The SP register is auto-decremented by 2. The Program Counter is then loaded

from the memory word addressed by the contents of the SP register, and the

program executes from that point.

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit: Not Affected.

Encodings: Addressing Mode Dest Word Opcode Length (none) Memory 3C 1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(none)	6	1

Symbolic	Encoding	Illustrative Exec. Time
RET	3C	8

[label:] RETI

Operation:

 $SP \leftarrow SP - 2$

PC ← word pointed to by SP

ENIR bit GIE ← 1

Description: The SP register is auto-decremented by 2. The Program Counter is then loaded from the memory word addressed by the contents of the SP register. The bit GIE in the ENIR register is then set (enabling maskable interrupts), and the program executes from the address given by the new contents of the Program Counter.

> To return from the Non-Maskable interrupt (NMI), one should test the CGIE bit in the PSW register. If it is set, RETI should be used to return. If it is not set, this means that interrupts were disabled when the NMI occurred, and a simple RET instruction should be used to return, keeping

interrupts disabled.

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit:

Not Affected.

Encodings

Addressing Mode	Dest	Wo Opcode	
(none)	Memory	3E	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(none)	6	1

Symbolic	Encoding	Illustrative Exec. Time
RETI	3E	8

RETSK Return from Subroutine and Skip

Syntax: [label:] RETSK

Operation: $SP \leftarrow SP - 2$

PC ← word pointed to by SP

Instruction pointed to by PC is skipped (not executed).

Description: The SP register is auto-decremented by 2. The Program Counter is then loaded

from the memory word addressed by the contents of the SP register. The instruction pointed to by the PC is skipped, and the following instruction is the

first executed.

Attributes: No-Address (see Section 5.2).

Size: None.

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Dest	Wo Opcode	ord Length
(none)	Memory	3D	1*

^{*} For timing purposes (only), use "2" as this value. See Section 5.2.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(none)	10	1

Symbolic	Encoding	Illustrative Exec. Time
RETSK	3D	12

[label:] RLC A

Operation:

 $C \leftarrow A[15] \leftarrow ... \leftarrow A[0] \leftarrow C$

Description: The contents of the accumulator are shifted left by one bit. The least-significant bit of the accumulator is loaded from the Carry bit, and the Carry bit receives the original contents of the most-significant bit of the accumulator. For example:

Before:

0011 1111 0000 0000

C = 1;

After:

0111 1110 0000 0001

C = 0

Attributes:

No-Address (see Section 5.2).

Size:

word (see Section 5.2).

Carry Bit:

Loaded from original contents of bit 15 of the accumulator.

Encodings:

Addressing Mode	Dest	Wo Opcode	ord Length
(None)	Accumulator	F7	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses	
(None)	3	0	

Symbolic	Encoding	Illustrative Exec. Time
RLC A	F7	4

$RRC\ A$ Rotate Accumulator Right Through Carry

Syntax: [label:] RRC A

Operation: $C \rightarrow A[15] \rightarrow ... \rightarrow A[0] \rightarrow C$

Description: The contents of the accumulator are shifted right by one bit. The most-significant

bit of the accumulator is loaded from the Carry bit, and the Carry bit receives the

original contents of the least-significant bit of the accumulator. For example:

Before: 0011 1111 0011 1111 C = 0;

After: 0001 1111 1001 1111 C = 1.

Attributes: No-Address (see Section 5.2).

Size: word (see Section 5.2).

Carry Bit: Loaded from original contents of bit 0 of the accumulator.

Encodings: Addressing Mode Dest Word Opcode | Length

(None) Accumulator D7 1

Timing Information:

Addressing Mode Min. Exec. Time Accesses

(None) 3 0

Examples:

Symbolic Encoding Illustrative Exec. Time

RRC A D7 4

[label:] SBIT n, dest

[label:] SBIT X, [B].B

Operation:

bit n of dest $\leftarrow 0$

Description: This instruction sets a bit in memory.

In the first form listed above, "n" is a constant in the range 0-7. The bit at this position in the destination byte operand is accessed.

The second form represents a specialized form of addressing allowed in the bit instructions IFBIT, RBIT and SBIT (see Section 4.4.3). Register B points to a byte in memory. Register X contains a bit offset in the range of 0-65535, which is applied relative to bit 0 of this byte, thus allowing any bit within 8K bytes of that byte to be accessed by this instruction. More formally, the byte address and bit position for the accessed bit are calculated as follows:

Address = B + (X + 8)

Bit No. = X modulo 8

where "B" and "X" represent the contents of the B and X registers.

Attributes:

One-Address (see Section 5.2).

Size: byte (see Section 5.2).

Carry Bit:

Not Affected.

Encodings:

A 11	Destination	Byte		
Addressing Mode	Destination	Opcode	Length	
Reg. B indirect	Memory Bit	08-0F	1	
Reg. X indirect	Memory Bit	08-OF 2		
Direct	Memory Bit	08-OF 3, 4 ¹		
Indirect	Memory Bit	08-0F	3*	
Indexed	Memory Bit	08-0F 4,5		
Reg. B/X Indirect	Memory Bit	39	1	

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	5	2
Reg. X indirect	8	2
Direct Basepage Non-Basepage	9 11	2/0
Indirect	11	3/2
Indexed 8-bit displacement 16-bit displacement	14 16	3/2 3/2
Reg. B/X Indirect	12	2

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
SBIT 7,[B].B	Reg. B Indirect	0F	8
SBIT 3,[X].W	Reg. X Indirect	8F 0B	12
SBIT 2, 38.W	Direct	96 26 0A	12
SBIT 5,[H'66].B	Indirect	AD 66 0D	16
SBIT 0,6[PNTR].W	Indexed	A2 06 88 08	18
SBIT X,[B].B	Reg. B/X Indirect	39	15

[label:] SC

Operation:

 $C \leftarrow 1$

Description: The Carry bit is set to a one.

Attributes:

No-Address (see Section 5.2).

Size:

Not Applicable.

Carry Bit:

Set to one.

Encodings:

Addressing Mode	Dest Word Opcode Le		
(None)	Carry bit	02	1

Timing

Information:

Addressing Mode	Min. Exec. Time	Accesses	
(None)	2	0	

Symbolic	Encoding	Illustrative Exec. Time
SC	02	3

SHL A Shift Accumulator Left

Syntax: [label:] SHL A

Operation: $C \leftarrow A[15] \leftarrow ... \leftarrow A[0] \leftarrow 0$

Description: The contents of the accumulator are shifted left by one bit. The least-significant

bit of the accumulator is cleared, and the Carry bit receives the original contents

of the most-significant bit of the accumulator. For example:

Before: $0011\ 1111\ 0000\ 0000$ $C = either\ 0 \text{ or } 1;$

After: $0111\ 1110\ 0000\ 0000$ C = 0.

Attributes: No-Address (see Section 5.2).

Size: word (see Section 5.2).

Carry Bit: Loaded from original contents of bit 15 of the accumulator.

Encodings: Addressing Mode Dest Word Opcode Length

(None) Accumulator E7 1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses	
(None)	3	0	

Symbolic	Encoding	Illustrative Exec. Time	
SHL A	E7	4	

Syntax: [label:] SHR A

Operation: $0 \rightarrow A[15] \rightarrow ... \rightarrow A[0] \rightarrow C$

Description: The contents of the accumulator are shifted right by one bit. The most-significant

bit of the accumulator is cleared, and the Carry bit receives the original contents

of the least-significant bit of the accumulator. For example:

Before: 0011 1111 0011 1111

C = either 0 or 1;

After:

0001 1111 1001 1111

C = 1.

Attributes: No-Address (see Section 5.2).

Size: word (see Section 5.2).

Carry Bit: Loaded from original contents of bit 0 of the accumulator.

Encodings

Addressing Mode	Dest	Word Opcode Length	
(None)	Accumulator	C7	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses	
(None)	3	0	

Symbolic	Encoding	Illustrative Exec. Time
SHR A	C7	4

ST Store Accumulator

Syntax: [label:] ST A, dest

Operation: A → dest

Description: The value in the accumulator is stored in the destination operand.

Attributes: One-Address (see Section 5.2).

Sizes: byte, word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Destination	Byte		Word	
Addressing Wode	Description	Opcode	Length	Opcode	Length
Reg. B indirect	Memory	C6	1	E6	1
Reg. X indirect	Memory	D6	1	F6	1
Direct	Memory				
Basepage		8B	2	AB	2
Non-Basepage		8B	4*	AB	4*
Indirect	Memory	8B	3*	AB	3*
Indexed	Memory	8B	4, 5*	AB	4, 5*

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	4	1
Reg. X indirect	4	1
Direct		
Basepage	7	1/0
Non-Basepage	11	1
Indirect	11	2/1
Indexed		
8-bit displacement	14	2/1
16-bit displacement	16	2/1

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
ST A,[B].B	Reg. B Indirect	C6	6
ST A,[X].W	Reg. X Indirect	F6	6
ST A, 38.W	Direct	8B 24	9
ST A,[H'66].B	Indirect	AD 66 8B	15
ST A, 6[PNTR].W	Indexed	A2 06 88 AB	19

SUBC

Subtract with Carry

[label:] SUBC dest, src Syntax:

 $dest \leftarrow dest - src - \overline{C}$ Operation:

C ← carry

Description: The source and destination operands are subtracted, and the result is placed in the destination operand. The Carry bit is incorporated into this result: if it is zero, the result is less by one than the difference of the operands; if it is one, the result is the exact difference. The Carry bit is then loaded with the carry from the subtraction (see Sections 2.3 and 2.4.2).

> This form of subtraction is intended to support multiple-precision arithmetic. For this use, the carry bit is first set (using the SC instruction), then SUBC is used to subtract the portions of the multiple-precision values, from least-significant to most-significant. Note that this is the only binary subtraction instruction in the HPC instruction set. To perform single-precision two's-complement subtraction, the Carry bit should always be set before executing this instruction.

Two-Address (see Section 5.2). Attributes:

byte→byte, word→word, byte→word (see Section 5.2).

Loaded with the carry from the subtraction (See Section 2.3 and 2.4.2): Carry Bit:

indicates that a borrow condition occurred;

indicates that no borrow condition occurred.

Encodings:

Addressine Mode	Destination	Byte :	Byte Source		Word Source	
Addressing Mode	Destination	Opcode	Length	Opcode	Length	
Reg. B indirect	Accumulator (16 bits)	СВ	1	EB	1	
Reg. X indirect	Accumulator (16 bits)	СВ	2*	EB	2*	
Direct	Accumulator (16 bits)	СВ	3, 4*	EB	3, 4*	
Indirect	Accumulator (16 bits)	СВ	3*	EB	3*	
Indexed	Accumulator (16 bits)	СВ	4, 5*	EB	4, 5*	
Immediate	Accumulator (16 bits)	EB	4*†	EB	5*†	
		Byte Dest.		Word	Dest.	
Direct-direct	Direct memory	СВ	4-6*	EB	4-6*	
Immediate-direct	Direct memory	СВ	4, 5*	EB	4-6*	

- * indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.
- † indicates addressing modes that are generated by the assembler as the appropriate immediate-to-direct forms (byte-word or word-word), using the fact that the accumulator is memory-mapped at address 00C8.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	4	1
Reg. X indirect	7	1
Direct Source in Basepage Source Non-Basepage	8 10	1/0
Indirect	10	2/1
Indexed 8-bit displacement 16-bit displacement	13 15	2/1 2/1
Immediate 8-bit immediate 16-bit immediate	13 15	3/0* 3/0*

	Dest. in B	asepage	Dest. Non-Basepage	
Addressing Mode	Min. Exec. Accesses		Min. Exec. Time	Accesses
Direct-Direct				
Source in Basepage	15	4/0*	. 17	4/3*
Source Non-Basepage	17	4/1*	19	4*
Immediate-Direct				
8-bit immediate	13	3/0*	15	3*
16-bit immediate	15	3/0*	17	3*

^{*} The dest operand is fetched twice; hence the extra access.

Symbolic	Addr. Mode	Carry From Bit	Encoding	Illustrative Exec. Time
SUBC A,[B].B	Reg. B Indirect	7	СВ	6
SUBC A.[X].W	Reg. X Indirect	15	8F EB	10
SUBC A, 38.W	Direct	15	96 26 EB	11
SUBC A, [H'66].B	Indirect	7	AD 66 CB	14
SUBC A, 6[PNTR].W	Indexed	15	A2 06 88 EB	18
SUBC A, #H'3CF2	Immediate-Direct	15	86 3C F2 C8 EB	20
SUBC BASEW, HIMEMB	Direct-Direct	15	84 66 77 AA EB	23
SUBC BASEB, #100	Immediate-Direct	7	82 64 55 CB	17

[label:] SWAP A

Operation:

 $A[11-15] \leftarrow A[8-11] \leftarrow A[4-7] \longleftrightarrow A[0-3]$

Description: The accumulator is shifted left four binary places (one "nibble"), and the loworder four bits of the accumulator are replaced with the original contents of bits 4-7. An example:

Before:

0101 0000 1111 1100

After:

0000 1111 1100 1111

Note that the two nibbles of the low-order byte have been swapped by this operation, and that bits 4-7 have been replicated in the low nibble of the highorder byte.

Attributes:

No-Address (see Section 5.2).

Size:

word (see Section 5.2).

Carry Bit:

Not Affected.

Encodings:

Addressing Mode	Dest	Wo Opcode	
(None)	Accumulator	3B	1

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
(None)	6	0

Symbolic	Encoding	Illustrative Exec. Time	
SWAP A	3B	7	

X Exchange with Accumulator

Syntax: [label:] X A, dest

Operation: $A \leftarrow \rightarrow dest$

Syntax: [label:] X A, [X+].B

[label:] X = A, [X+].W[label:] X = A, [X-].B[label:] X = A, [X-].W

Operation: $A \leftarrow \rightarrow$ location pointed to by X

 $X \leftarrow X \pm 1 \text{ or } 2$

Description: The contents of the accumulator and the destination operand are exchanged. Note that registers as well as memory locations may be accessed with this instruction

by using their memory-mapped addresses.

In addition to the general one-address addressing modes, the X instruction has forms allowing use of X-register indirect addressing with auto-increment or

auto-decrement.

See also the XS instruction, which performs a B-register indirect access with auto-

increment or auto-decrement and a conditional skip.

Attributes: One-Address (see Section 5.2).

Specialized addressing extension: X-Register Indirect with auto-increment or

auto-decrement.

Sizes: byte, word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Destination	Byte Opcode	Dest Length	Word Opcode	Dest Length
Reg. B indirect	Accumulator	C5	1	E5	1
Reg. X indirect	Accumulator	D5	1	F5	1
Direct Basepage Non-Basepage	Accumulator	8E 8E	2 4*	AE AE	2 4*
Indirect	Accumulator	8E	3*	AE	3*
Indexed	Accumulator	8E	4, 5*	AE	4, 5*
X-Indirect Auto-Inc. X-Indirect Auto-Dec.	Accumulator Accumulator	D1 D3	. 1	F1 F3	1 1

indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Reg. B indirect	5	2
Reg. X indirect	5	2
Direct Dest in Basepage Dest Non-Basepage	7 11	2/0 2
Indirect	11	3/2
Indexed 8-bit displacement 16-bit displacement	14 16	3/2 3/2

Addressing Mode	Min. Exec. Time	Accesses
Reg. X indirect, Auto-Inc.	5	2
Reg. X indirect, Auto-Dec.	5	2

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
X A,[B].B	Reg. B Indirect	C5	8
X A,[X].W	Reg. X Indirect	F5	8
X A, 38.W	Direct	AE 26	9
X A,[H'66].B	Indirect	AD 66 8E	16
X A, 6[PNTR].W	Indexed	A2 06 88 AE	20
X A,[X+].W	X-Indir. Auto-Inc.	F1	8
X A, [X-].B	X-Indir. Auto-Dec.	D3	8

Syntax: [label:] XOR dest, src

Operation: dest ← dest XOR src

Description: This instruction performs a bitwise Exclusive OR operation between the bits of

the destination and source operands, and places the result into the destination. In byte-to-word cases, the upper byte of the destination is unaffected due to

automatic zero-extension of the source (see Section 2.4.2).

Attributes: Two-Address (see Section 5.2).

Sizes: byte→byte, word→word, byte→word (see Section 5.2).

Carry Bit: Not Affected.

Encodings:

Addressing Mode	Destination	Byte Source		Word Source	
		Opcode	Length	Opcode	Length
Reg. B indirect	Accumulator (16 bits)	DB	1	FB	1
Reg. X indirect	Accumulator (16 bits)	DB	2*	FB	2*
Direct	Accumulator (16 bits)	DB	3, 4*	FB	3, 4*
Indirect	Accumulator (16 bits)	DB	3*	FB	3*
Indexed	Accumulator (16 bits)	DB	4, 5*	FB	4, 5*
Immediate	Accumulator (16 bits)	9B	2	BB	3
		Byte Dest.		Word Dest.	
Direct-direct	Direct memory	DB	4-6*	FB	4-6*
Immediate-direct	Direct memory	DB	4, 5*	FB	4-6*

^{*} indicates that an addressing directive (Section E.2) is used as a prefix to the opcode; the prefix establishes the length of the instruction.

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses 1 1	
Reg. B indirect	4		
Reg. X indirect	7		
Direct			
Source in Basepage	8	1/0	
Source Non-Basepage	10	1	
Indirect	10	2/1	
Indexed			
8-bit displacement	12	2/1	
16-bit displacement	14	2/1	
Immediate			
8-bit immediate	4	0	
16-bit immediate	6	0	

	Dest. in B	asepage	Dest. Non-Basepage		
Addressing Mode	Min. Exec. Time	Accesses	Min. Exec. Time	Accesses	
Direct-Direct					
Source in Basepage	13	3/0	15	3/2	
Source Non-Basepage	15	3/1	17	3	
Immediate-Direct					
8-bit immediate	11	2/0	13	2	
16-bit immediate	13	2/0	15	2	

Symbolic	Addr. Mode	Encoding	Illustrative Exec. Time
XOR A,[B].B	Reg. B Indirect	DB	6
XOR A,[X].W	Reg. X Indirect	8F FB	10
XOR A, 38.W	Direct	96 26 FB	11
XOR A,[H'66].B	Indirect	AD 66 DB	14
XOR A, 6[PNTR].W	Indexed	A2 06 88 FB	17
XOR A, #H'3CF2	Immediate	BB 3C F2	9
XOR BASEW, HIMEMB	Direct-Direct	84 66 77 AA FB	21
XOR BASEB, #100	Immediate-Direct	82 64 55 DB	15

XS

Exchange with A, with Auto-Increment/Decrement and Conditional Skip

Syntax:

[label:] XS [B+].B[B+].W [label:] XS [label:] XS [B-].B

[B-].W [label:] XS

Operation:

 $A \leftarrow \rightarrow$ operand (pointed to by B) $B \leftarrow B \pm \text{ operand's width (1 or 2)}$

If B incremented/decremented past address in K, skip next instruction.

Description: The contents of the location addressed by the B register are exchanged with those of the accumulator. The B register is then auto-incremented or auto-decremented (as selected by either "+" or "-") by the number of bytes in the operand (i.e., 1 for the Byte form, or 2 for the Word form of the instruction). If after incrementing B the result is greater than the value in the K register, the following instruction is skipped (not executed). If after decrementing B the result is less than the value in the K register, the following instruction is skipped.

> Note that, since the HPC is an unsigned machine, no value is considered greater than FFFF Hex or less than zero; the K register may therefore be set to one of these values to effectively disable skipping. By the same token, however, neither decrementing while K = 0 nor incrementing while K = FFFF will ever cause a skip (for example, to terminate a loop). Loops involving the first or last byte (or word) of memory, then, should use this instruction with care.

Attributes:

Specialized addressing: using Register B (see Section 4.4.2).

Sizes: byte, word (see Section 5.2).

Carry Bit:

Not Affected.

Encodings:

Addressing Mode	Source	Byte Opcode Length		'''	ord Length	
Auto-Increment	Memory	C1	1	E1	1	
Auto-Decrement	Memory	С3	1	E3	1	

Timing Information:

Addressing Mode	Min. Exec. Time	Accesses
Without Skip	5	2
With Skip	9+*	2

* This value must be increased by one for each Wait state imposed on instruction fetches. See Sections 5.2 and 10.4.

Examples:

Symbolic	Conditions	Encoding	Illustrative Exec. Time
XS A,[B+].B	Auto-Increment by 1, No Skip	C1	8
XS A, [B-].W	Auto-Decrement by 2, Skipping	E3	13

		\
		\
		\
		**.

FUNCTIONAL PIN DESCRIPTION

6.1 HPC16083 PIN DESCRIPTION

As is required in any circuit designed with MOS transistors, the logic on the HPC16083 is protected from damage by electrostatic discharges. Diode protection is provided on all pins of the device. The HPC16083 uses 68 pins for interface to the user environment (see Figures 6-1 and 6-2). Refer to the data sheet for the latest packaging information.

I/O Port Pins

- A0—A15 comprise Port A, which is a 16-bit bidirectional I/O port with a Data Direction register (DIRA), allowing each separate pin to be individually defined as an input or output (see Section 9.2). When the Universal Peripheral Interface (UPI) feature is enabled (Chapter 19), Port A becomes a bus connection by which a host processor can transfer data, status and commands to and from the HPC software. When the HPC16083 is configured to access external memory, Port A is used as the multiplexed address/data bus (Section 10.1).
- B0—B15 comprise Port B, which is a 16-bit bidirectional I/O port with a structure similar to Port A. Port B has associated with it a Data Direction register DRB and a Function register BFUN to individually allow each pin to serve an alternate function as detailed below. See also Section 9.3.

```
B0:
      TDX
                 - UART Data Output
                 — (none)
B1:
                 — UART Clock (Input or Output)
B2:
      CKX
B3:
      T2IO
                 - Timer T2 I/O pin
                 - Timer T3 I/O pin
      T3IO
B4:
                 - MICROWIRE/PLUS Output
      SO
B5:
                 — MICROWIRE/PLUS Clock (Input/Output)
B6:
      SK
      HLDA
                 - Hold Acknowledge Output
B7:
       TS<sub>0</sub>
                 — Timer Synchronous Output
B8:
       TS1
                 - Timer Synchronous Output
B9:

    Address 0 Input for UPI Mode

B10:
       UA0
       WRRDY

    Write Ready Output for UPI Mode

B11:
                  — (none)
B12:
                  - Timer Synchronous Output
B13:
       TS2
       TS3
                  - Timer Synchronous Output
B14:
                  - Read Ready Output for UPI Mode
B15:
       RDRDY
```

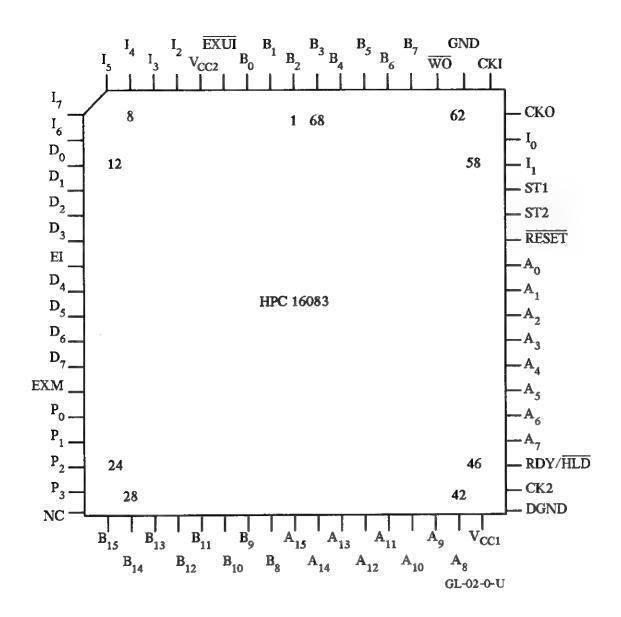
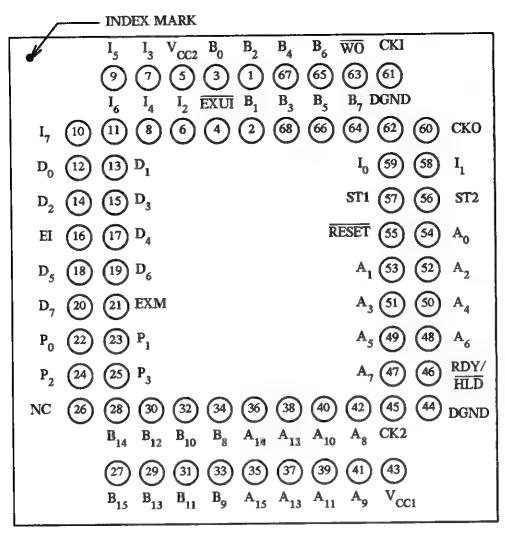


Figure 6-1. Pinout Diagram for HPC16083: 68-Pin PLCC and LCC Packages



GL-52-0-U

NOTE: This is also the top view pin-out of the HPC MOLE emulator cable.

Figure 6-2. Pinout Diagram for HPC16083: 68-Pin PGA Package

When the HPC16083 is configured to access external memory, four bits of Port B are used for bus control as follows. (This overrides any other function selected above.) See Section 10.1.

B10: ALE — Address Latch Enable Output

B11: \overline{WR} — Write Output

B12: HBE — High Byte Enable Output

B15: RD — Read Output

IO—I7 comprise Port I, which is an 8-bit input port that can be read as general purpose inputs and can also be used for the following functions (see Section 9.4):

I0: — (none)

II: NMI — Nonmaskable Interrupt Input

INT2 — Interrupt 2 / Input Capture 2 / UPI URD
 INT3 — Interrupt 3 / Input Capture 3 / UPI UWR

I4: INT4 — Interrupt 4 / Input Capture 4

I5: SI — MICROWIRE Data Input

I6: RDX — UART Data Input

17: — (none)

- D0—D7 comprise Port D, which is an 8-bit input port that is used for general purpose digital inputs. See Section 9.5.
- PO-P3 comprise Port P, which is a 4-bit output port that can be used as general purpose outputs, or its pins can be selected individually to be controlled by Timers T4 through T7 to generate constant frequency, variable duty cycle or pulse-width modulated outputs. See Section 9.5.

Power Supplies

VCC1 Positive power supply voltage

VCC2 Positive power supply voltage

GND Ground reference for on-chip logic

DGND Ground for on-chip drivers connected to output pins

The VCC1 and VCC2 pins are connected on-chip, but the GND and DGND pins are electrically isolated. See Section 7.1.

Clock Pins

CKI Oscillator Input

CKO Oscillator Output

CK2 Clock output (CKI divided by 2)

Pins CKI and CKO are usually connected across an external crystal. See Section 7.2.

Other Pins

- WO This is an open-drain output pin that signals, when low, detection of an illegal condition by the Watchdog logic. See Chapter 13.
- ST1 Indicates that the current read cycle is being executed to fetch in the first opcode byte of a new instruction. See Section 10.7.
- ST2 Indicates the instruction status in conjunction with the ALE signal. See Section 10.7.
- RESET is an active low input that forces the chip to restart. See Section 7.3.
- RDY/HLD has two uses, selected by bit 0 of the ENIR register. It is either a READY input, used to extend the bus cycle for slower memories, or a HOLD request input to float the bus for DMA purposes. The processor defaults to the HOLD interpretation of this pin. See Section 10.5 and Chapter 11.
- EXM External memory enable (active high). This pin disables the on-chip ROM and maps it to external memory, placing the HPC16083 in ROMless Mode (Section 10.1.2).
- EI External Maskable Interrupt input pin. May be programmed to be high/low level sensitive or rising/falling edge sensitive. El is also used as input for fourth input capture register (EICR). See Chapter 15.
- EXUI External UART Interrupt input pin: level-sensitive and low-active, with an internal pull-up device. See Section 18.6.
- NC No Connection; these pins are reserved for NSC internal or future use. MAKE NO CONNECTIONS to these pins.

		_
		`
		`

HARDWARE CONNECTIONS

7.1 POWER AND GROUNDING

The HPC16083 requires a single power supply, applied on the VCC1 and VCC2 pins (see Figure 7-1). This voltage is nominally 5 volts, but can be reduced as explained in the data sheet.

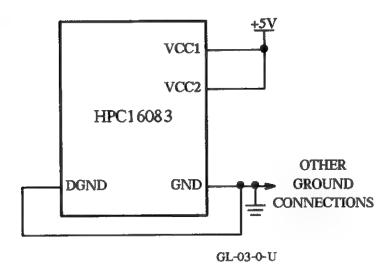


Figure 7-1. Power Supply Connections

Grounding connections are made to two pins. Logic Ground (GND) is the common ground reference for on-chip logic, and Driver Ground (DGND) is the common ground reference for the on-chip output buffers. If a ground plane is not being used, DGND should be connected through a single conductor directly to GND, and all other connections should be made only to GND.

7.2 CLOCKING

The on-chip clocking circuitry is illustrated in Figure 7-2. It consists of a phase inverter connected between the pins CKI (Clock In) and CKO (Clock Out) which serves as an oscillator, and a divide-by-two stage, which provides internal clocking and the signal on the CK2 (Clock/2) pin.

Figure 7-3 shows the recommended crystal network. Crystals should be specified as AT cut for parallel resonant operation at the desired load capacitance.

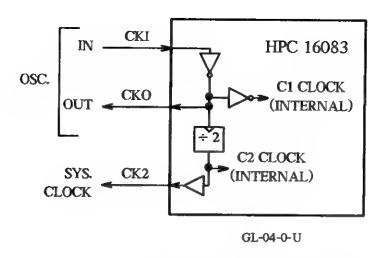


Figure 7-2. On-Chip Clocking Circuit

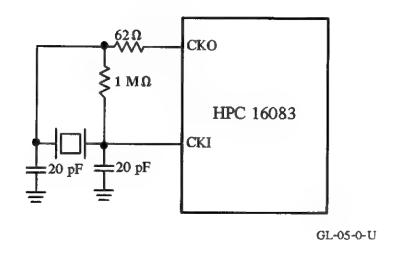


Figure 7-3. Typical Crystal Oscillator Connections

Generating the clock with an external oscillator can be done by driving the CKI pin as shown in Figure 7-4. The duty cycle of this signal is nominally 50%, with limits as defined in the data sheet. Note that the CKO output signal is inverted from the CKI input, and that the CK2 clock toggles on rising edges of CKO (i.e., falling edges of CKI).

* Resistor not necessary
if HCMOS driver is used.
For TTL drivers, resistor
should be 500 Ω or less.

CKI

CKI

CKO

HPC 16083

CK2

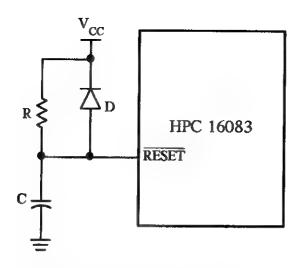
Figure 7-4. External Clocking Connections

7.3 RESET

The RESET pin serves to restart the on-chip logic. The effects of this are detailed in Chapter 7.3. The HPC can be reset at any time by asserting the RESET pin low for at least 16 cycles of the CK2 clock. In applications using the Watchdog feature, RESET should be asserted for at least 64 cycles of the CK2 clock. This is explained in Chapter 13.

On application of power, RESET must be held low for at least five times the power supply rise time to ensure that the on-chip oscillator circuit has time to stabilize.

A simple technique for generating a power-on reset is to connect an external RC network to the RESET pin as illustrated in Figure 7-5.



 $RC \ge 5 \times POWER SUPPLY RISE TIME$

GL-07-0-U

Figure 7-5. Power-On Reset Circuit

MEMORY ORGANIZATION AND REGISTER SET

8.1 INTRODUCTION

The HPC16083 has a total address space of 64 Kbytes, most of which is available to the user for both program and data memory. Both ROM and ROMless part versions are available, with up to 8 Kbytes of ROM and 256 bytes of RAM present on chip. Both ROM and RAM share the same address space, thus allowing instructions to be executed out of RAM. Program memory addressing is done on a byte basis by the 16-bit program counter. Memory can be addressed directly by instructions or indirectly through the B, X, or SP registers (Sections 3.1 and 8.4). Data memory can be addressed as words or bytes, with words being addressed on even byte boundaries. The device uses memory-mapped organization to support access to registers, I/O and on-chip peripheral functions.

8.2 MEMORY MAP

The memory map for the HPC16083 is as shown in Table 8-1.

8.3 CONTROL REGISTERS

All of the I/O on the HPC16083 is configured by the control registers. The memory map above gives the names and locations of these registers and Table 8-2 lists all of the control registers and indicates where additional information can be found in this manual.

8.4 DEDICATED REGISTERS

There are six 16-bit registers on the HPC16083 whose roles are dedicated to software. These registers have the following functions:

- A The A register (Accumulator) is the source and destination register for most I/O, arithmetic, logic and memory data access operations.
- B, X The two registers B and X can be used for indirect addressing. They can automatically count up and down to sequence through memory data.
- K The K register is used to set limits for repetitive loops of code as register B sequences through memory data.

TABLE 8-1. MEMORY MAP OF HPC16083

	r	
FFFF:FFF0	Interrupt Vectors in on-chip ROM	
FFEF:FFD0	JSRP Vectors in on-chip ROM	USER MEMORY
FFCF:FFCE		
:	On-Chip ROM	
E001:E000	on omp nom	
DFFF:DFFE		
:	External Expansion	
0201:0200	Memory	
01FF:01FE		
•	On-Chip RAM	USER RAM
01C1:01C0	_	
0195:0194	Watchdog Register	Watchdog Logic
0192	TOCON Register	
0191:0190	TMMODE Register	
018F:018E	DIVBY Register	
018D:018C	T3 Timer	
018B:018A	R3 Register	
0189:0188	T2 Timer	Timer Block T0:T3
0187:0186	R2 Register	
0185:0184	I2CR Register/ R1	
0183:0182	I3CR Register/ T1	
0181:0180	I4CR Register	
015F:015E	EICR Register	
015C	EICON Register	
0153:0152	PORTP Register	
0151:0150	PWMODE Register	
014F:014E	R7 Register	
014D:014C	T7 Timer	
014B:014A	R6 Register	
0149:0148	T6 Timer	Timer Block T4:T7
0147:0146	R5 Register	
0145:0144	T5 Timer	
0143:0142	R4 Register	
0141:0140	T4 Timer	
I		ı

TABLE 8-1. (Cont)

	1	ı
0128	ENUR Register	
0126	TBUF Register	
0124	RBUF Register	UART
0122	ENUI Register	
0120	ENU Register	
0104	PORTD Input Register	PORT D
00F5:00F4	BFUN Register	
00F3:00F2	DIRB Register	PORTS A & B CONTROL
00F1:00F0	DIRA Register / IBUF	
00E6	UPIC register	UPI CONTROL
00E3:00E2	PORTB	
00E1:00E0	PORTA Register / OBUF	PORTS A & B
OODE	Microcode ROM Dump	
00DD:00DC	Halt Enable Register	
00D8	PORTI Input Register	PORT L CONTROL &
00D6	SIO Register	INTERRUPT CONTROL
00D4	IRCD Register	REGISTERS
00D2	IRPD Register	
00D0	ENIR Register	
00CF:00CE	X Register	
00CD:00CC	B Register	
OOCB:OOCA	K Register	
00C9:00C8	A Register	HPC CORE
00C7:00C6	PC Register	REGISTERS
00C5:00C4	SP Register	
00C3:00C2	(reserved)	
00C0	PSW Register	
OOBF:OOBE	On-Chip	USER RAM
•	RAM	
0001:0000		

TABLE 8-2. CONTROL REGISTERS

REGISTER	DESCRIPTION	SECTION
PSW	Processor Status Register	8.5
DIRA	Port A Direction Register	9.2
PORTA	Port A Data Register	9.2
DIRB	Port B Direction Register	9.3
PORTB	Port B Data Register	9.3
BFUN	Port B Alternate Function Register	9.3
PORTI	Port I Data Input Register	9.4
PORTD	Port D Data Input Register	9.5
PORTP	Port P Data and Control Register	9.6
WATCHDOG	Watchdog Timer Register	Chapter 13
HALT	Halt Enable Control Register (at 00DC)	14.2
IRCD	Interrupt/Input Capture Condition Register	15.5
ENIR	Interrupt Enable Register	15.5
IRPD	Interrupt Pending Register	15.5
TOCON	Timer TO Configuration Register	16.4
EICON	El Configuration Register	16.4
TMMODE	Configuration Register for Timers TO - T3	16.4
PWMODE	Configuration Register for Timers T4 - T7	16.4
DIVBY	Timer Rate Control Register	16.4
SIO	MICROWIRE/PLUS Serial Input/Output Register	17.2
TBUF	UART Transmitter Buffer Register	18.3
RBUF	UART Receiver Buffer Register	18.3
ENUR	UART Receive Control and Status Register	18.9
ENU	UART Control and Status Register	18.9
ENUI	UART Interrupt and Clock Source Register	18.9
IBUF	UPI Data Input Register (DIR A Register used as IBUF when in UPI Mode)	19.2
OBUF	UPI Data Output Register (PORT A Register used as OBUF when in UPI Mode)	19.2
UPIC	UPI Control Register	19.3

SP This is a pointer which addresses the stack. The stack is used for storing temporary data, and holding return information for subroutines and interrupt service routines. The SP register points to the next available (empty) location on the stack.

The HPC stack grows upward in memory. A push operation post-increments the stack by two bytes, and a pop operation pre-decrements the stack pointer by two bytes.

PC The Program Counter register is a pointer, used by the HPC to address memory while fetching instructions. It is a read-only register when accessed at its memory-mapped address (00C6).

8.5 PROCESSOR STATUS WORD (PSW)

The PSW register is an 8-bit register whose format is detailed below.

PSW	**	CGIE	C	EA	WAIT1	WAITO	HLT/ ID L	EHI
	X	XR	X	0	0	0	0	0
				Byte a	t 00C0			

0 = Cleared on Reset

X = State Indeterminate on Reset

R = Read-Only

Bit Descriptions

##

Unassigned bit. Should be considered indeterminate data.

CGIE

This bit is loaded from the GIE bit of the ENIR register whenever an interrupt begins service. It indicates, for purposes of non-maskable interrupt handling, whether interrupts should be re-enabled on return from the service routine. This is a read-only bit.

C

This bit indicates that a carry or borrow occurred during an addition or subtraction instruction. When adding, a one indicates a carry and a zero indicates no carry. When subtracting, a zero indicates a borrow and a one indicates no borrow. In shifting instructions, the C bit receives the bit shifted out of the accumulator. In division instructions, the C bit is used as an error flag.

EA

This bit, External Access, defines the allowable addressing range, thus configuring the HPC to operate in Single Chip mode or Expanded mode (see Section 10.1.1). Resetting this bit restricts the allowed addressing to the on-chip address range only, while setting it allows the full 64 Kbytes of addressing space to be used.

WAIT1/WAIT0

These two bits allow the number of Wait states (Section 10.4) to be programmed. Encodings are:

00 -- 4 Wait states

01 - 2 Wait states

10-1 Wait state

11 - No Wait states

HLT/IDL

The HLT/IDL mode select bit (see Chapter 14). This bit works in conjunction with the EHI PSW bit to select either the HALT or the IDLE power save mode. HALT mode is selected by setting this bit, and IDLE mode by resetting it.

EHI

This control bit works in conjunction with the HLT/ $\overline{\text{IDL}}$ control bit, and setting this bit through software causes the HPC to enter HALT or IDLE mode, as selected by the HLT/ $\overline{\text{IDL}}$ bit.

PORT STRUCTURES AND OPERATION

9.1 INTRODUCTION

The user-programmable pins of the HPC16083 are divided into five ports, designated A, B, D, I and P. This chapter details the operation of each of these ports.

9.2 PORT A

Port A is a set of 16 bidirectional I/O pins, serving one of three purposes. It can be configured as:

- A bidirectional Address/Data bus for communication with off-chip memory or peripherals. This function is explained in Chapter 10.
- An 8-bit or 16-bit Universal Peripheral Interface (UPI) port, through which an external
 host processor can give commands to HPC on-chip programs, receive status indications, and
 transfer data. This function is explained in Chapter 19.
- 16 bit-programmable I/O pins, independently configurable as inputs or outputs. This function is explained in this section.

These three functions of Port A are mutually exclusive. By enabling External Access Mode via the PSW EA bit or ROMless Mode via the EXM pin, Port A becomes dedicated as an Address/Data bus. By setting the UPIEN bit in the UPIC register, one or both bytes of Port A become dedicated as a Data bus for the UPI function. Otherwise, the Port A pins may be used as bit-programmable I/O pins as described here.

Figure 9-1 shows the structure of Port A while it is being used as a bit-programmable I/O port. Associated with Port A are two read/write control registers accessible to software: a 16-bit Data register (PORTA, at addresses 00E0 and 00E1) and a 16-bit Direction register (DIRA, at addresses 00F0 and 00F1). Pin A0 corresponds to bit 0 (the least-significant bit) of these registers, pin A1 corresponds to bit 1, and so on. Software may access these registers as single 16-bit quantities, specifying the even address, or as separate bytes, specifying the even address for the least-significant byte and the odd address for the most-significant byte.

The DIRA register allows each of the Port A pins to be individually programmed as inputs or outputs.

Pins configured as outputs, by writing ones to the corresponding bits in DIRA, assume the values written into the PORTA data register, while reading these pins returns the value in the Data register.

Pins that are selected as inputs, by clearing the corresponding bits in the DIRA register, are caused to float. A Write access to a pin configured as an input causes the value to be written into the PORTA data register (having no effect unless the pin is subsequently changed to an output), and a Read access returns the state of the pin itself.

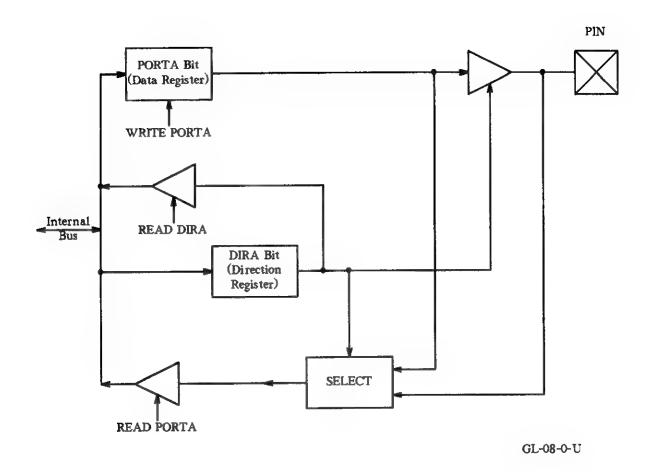
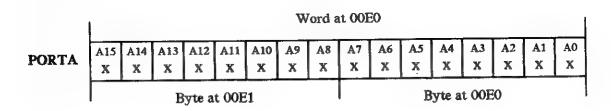
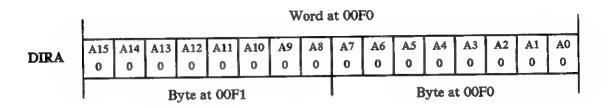


Figure 9-1. Structure of Port A

The bit maps below give, for each bit in the PORTA and DIRA registers, the pin affected by that bit and the state of that bit when the HPC is reset.





Cleared at reset.

X State on reset is indeterminate.

Note that all Port A pins are set floating on reset. For output functions in which a particular state must be present immediately on reset, a pull-up or pull-down resistor should be attached to the pin.

To correctly initialize Port A after a Reset, one should:

- Set all bits in the PORTA register to their proper initial states.
- Write ones to all bits in the DIRA register corresponding to output pins.

9.3 PORT B

Port B consists of 16 pins, 12 of which are bidirectional I/O similar in structure to Port A. The remaining four pins (B10, B11, B12 and B15) are outputs only; designating them as inputs causes them to float, but does not allow the states of these pins to be monitored by software.

Port B has a Data register (PORTB, at address 00E2), a Direction register (DIRB, at address 00F2) and an Alternate Function register (BFUN, at address 00F4) that configures individual Port B pins to take on dedicated alternate I/O functions. The PORTB and DIRB registers operate as described in the previous section for Port A, with the exception that bits 10, 11, 12 and 15 of DIRB serve only to float the corresponding Port B pins. Placing a one in a BFUN register bit causes the corresponding pin's alternate function to be selected.

The alternate functions provided on PORTB pins are:

B 0	_	TDX	UART Data Output	(see Section 18.2)
B 1	_	(none)	-	
B2	_	CKX	UART Clock (Input or Output)	(see Section 18.2)
В3		T2IO	Timer 2 Input or Output	(see Section 16.3)
B4		T3IO	Timer 3 Input or Output	(see Section 16.3)
B5	_	SO	MICROWIRE/PLUS Data Output	(see Chapter 17)
B6	*****	SK	MICROWIRE/PLUS Clock (Input or Output)	(see Chapter 17)
B7	-	HLDA	Hold Acknowledge Output	(see Chapter 11)
B8	_	TSO	Timer Synchronous Output	(see Section 16.3)
B9		TS1	Timer Synchronous Output	(see Section 16.3)
B10	_	UAO	UPI Port Address 0 Input	(see Chapter 19)
B11	_	WRRDY	UPI Port Write Ready Output	(see Chapter 19)
B12	_	(none)		
B13	_	TS2	Timer Synchronous Output	(see Section 16.3)
B14		TS3	Timer Synchronous Output	(see Section 16.3)
B15	_	RDRDY	UPI Port Read Ready Output	(see Chapter 19)

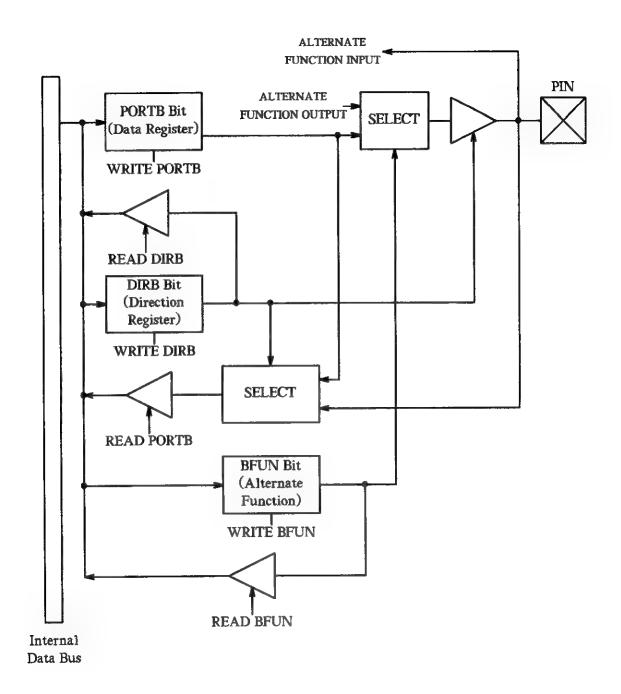
NOTES: 1. The CKX (B2) and UA0 (B10) pins do not require that their BFUN bits be set.

2. BFUN bit 6 (SK) should not always be set when using the MICROWIRE/PLUS feature. See Section 17.3.

The structure of a generic Port B pin is shown in Figure 9-2. Note that selecting the alternate function for the pin causes its output value to come from a separate source (other than the PORTB data register bit). There are two modifications to this generic scheme, as explained below and in Figures 9-3 and 9-4.

The Port B pins which perform as Timer Synchronous outputs when their alternate functions are selected (TSO-3, T2IO and T3IO) do not take their output values from a separate source (see Figure 9-3). Instead, while the corresponding BFUN bits are set, their data bits in the PORTB register are enabled to toggle on underflow from the appropriate timers.

When the Expanded or ROMless mode of operation is entered (see Section 10.1), the three Port B pins B10, B11 and B15 become the control bus signals ALE, WR and RD respectively. If the bus width selected at Reset (via a hardware strap option, Section 10.1.3) is 16 bits, pin B12 is also dedicated as the HBE signal; otherwise, this pin is available for bit-programmable output. These assignments override any assignments made in the corresponding BFUN register bits. The structure of these four pins is shown in Figure 9-4. Note that they are outputs only; reading from the PORTB register address returns the contents of the PORTB register bit regardless of the setting of the DIRB bit.



GL-09-0-U

Figure 9-2. Structure of Port B Pins B0, B1, B2, B5, B6 and B7 (Generic Structure)

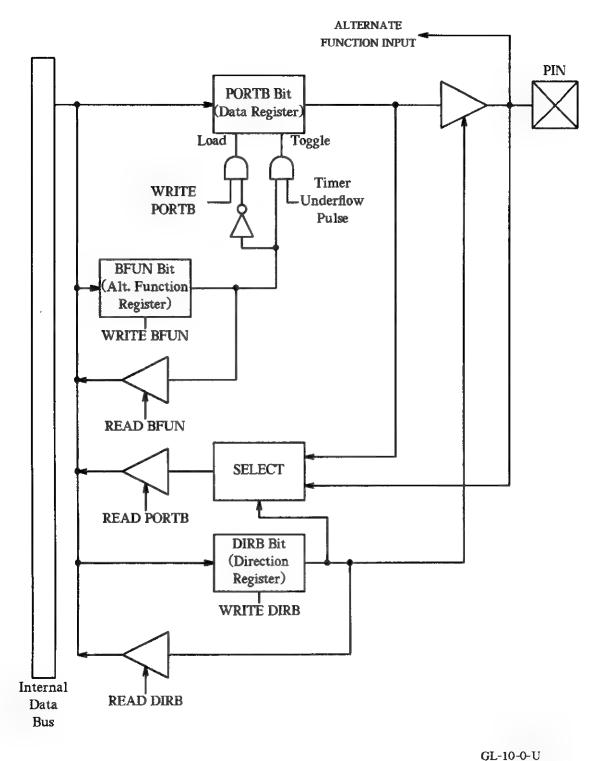
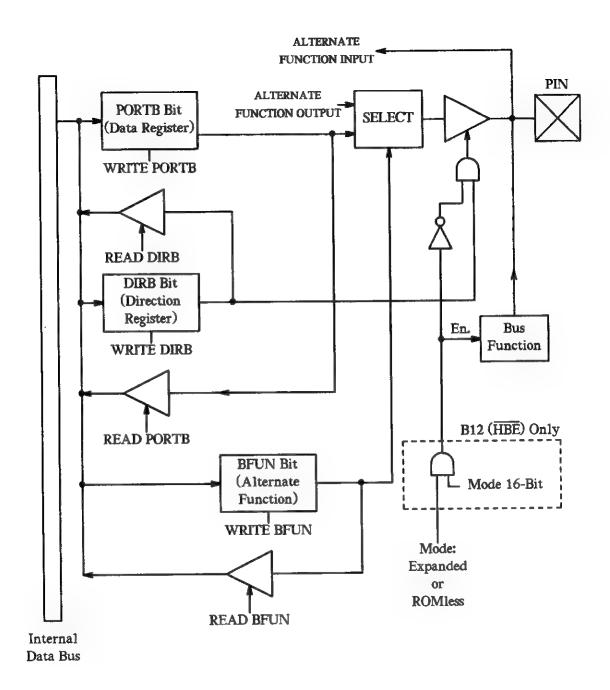


Figure 9-3. Structure of Port B Pins B3, B4, B8, B9, B13 and B14 (Timer Synchronous Pins)



GL-11-0-U

Figure 9-4. Structure of Port B Pins B10, B11, B12 and B15 (Pins with Bus Control Roles)

- NOTES: 1. While a Timer Synchronous function is enabled on a Port B pin (i.e., while its BFUN bit is a one), the PORTB register bit cannot be altered by software. This prevents accidental nullification of a timer-initiated toggle by software writes to other bits in the same byte of the PORTB register.
 - 2. Any Port B pin selected for an alternate function must also have its DIRB bit set appropriately for that function (i.e. the direction of the pin is not determined automatically by setting the BFUN bit).

	1	Word at 00E2															
PORTB	B15 Xo	B14 X	B13 X	B12 Xo	B11 Xo	B10 Xo	B9 X	B8 X	B7 X	B6 X	B5 X	B4 X	B3 X	B2 X	B1 X	B0 X	
	Byte at 00E3										Byte at 00E2						

,		Word at 00F2														
DIRB	B15	B14	B13	B12	B11	B10	B9	B8 0	B7	B6 0	B5	B4 0	B3 0	B2 0	B1 0	B0 0
	· ·		В	yte a	t 00F	3			Byte at 00F2							Ů

		Word at 00F4														
BFUN	RDR DY O	0	0 125	0	wrrdy O	0	TS1 O	T\$0 0	HILDA O	SIK O	\$0 0	T3 I/O	12 I/O O	CKX 0	‡ 0	TDX 0
			В	yte a	t 00F	5					Е	lyte a	t 00F	4		

- ** Unassigned bit. Should be considered indeterminate data.
- X State indeterminate at reset.
- O Cleared at reset.
- Output only. These Port B pins are never inputs; they will, however, float when the corresponding DIRB bits are cleared.

Note that all Port B pins are set floating on reset. For output functions in which a particular state must be present immediately on reset, a pull-up or pull-down resistor should be attached to the pin.

To correctly initialize Port B pins as bit-programmable I/O pins after a Reset, one should:

- Set all bits in the PORTB register to their proper initial states.
- Write ones to all bits in the DIRB register corresponding to output pins.

Initialization of Port B pins for their alternate functions is discussed in the chapters dealing with the on-chip features that use these functions.

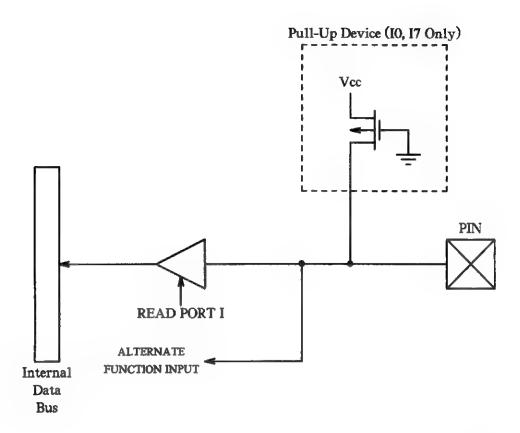
9.4 PORT I

The eight pins of Port I are inputs, as shown in Figure 9-5. They may be sampled by software by reading from the 8-bit PORTI location at address 00D8. They also serve alternate functions, as required to support on-chip features. There is no Alternate Function register for Port I; a feature, when activated, will automatically use the appropriate Port I pin for input. Input pin I1 is dedicated as the Nonmaskable Interrupt. Listed below are the functions that are available on the individual Port I pins.

Ю		No alternate function	
IO	_		(64 . 45)
I1	_	Nonmaskable Interrupt (NMI)	(see Chapter 15)
12	_	Interrupt 2 / Input Capture / URD	(see Chapter 15, Section 16.2
			and Chapter 19)
13	_	Interrupt 3 / Input Capture / UWR	(see Chapter 15, Section 16.2 and Chapter 19)
14	_	Interrupt 4 / Input Capture	(see Chapter 15, Section 16.2)
I 5	_	SI — MICROWIRE PLUS Serial Data Input	(see Chapter 17)
16		RDX — UART Serial Data Input	(see Section 18.2)
I7	_	No alternate function	

PORTI	I7	I6/RDX	15/SI	I4/INT4	I3/INT3	I2/INT2	NMI	IO			
	XR	XR	XR	XR	XR	XR	XR	XR			
	Byte at 00D8										

X State indeterminate at reset.R Read Only. Writing has no effect.



GL-12-0-U

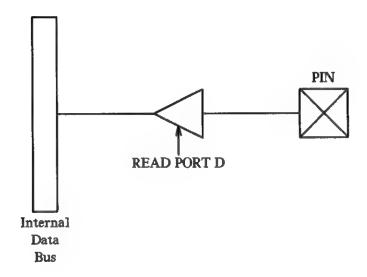
Figure 9-5. Structure of Port I

9.5 PORT D

Port D is a set of eight pins which can be used as general purpose digital inputs. They are sampled by reading the eight-bit PORTD location at address 0104 (hex). (See Figure 9-6.)

PORTD	D7	D6	D5	D4	D3	D2	D1	D0				
	XR	XR	XR	XR	XR	XR	XR	XR				
	Byte at 0104											

- X State indeterminate at reset.
- R Read Only. Writing has no effect.



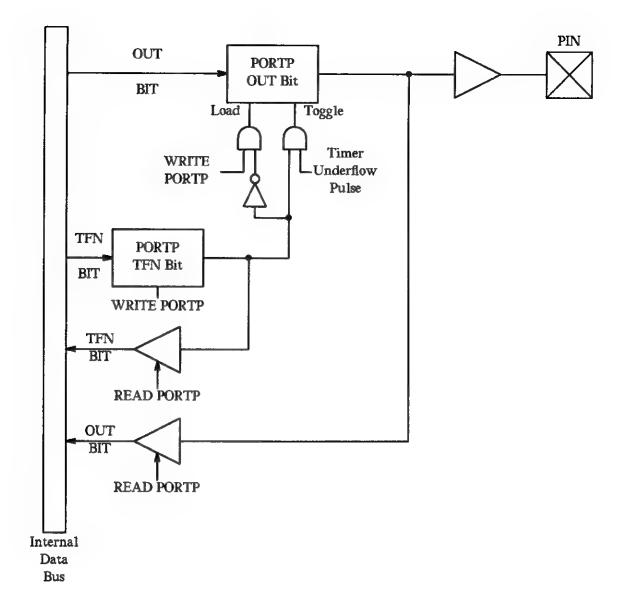
GL-13-0-U

Figure 9-6. Structure of Port D

9.6 PORT P

Port P is a set of four output pins that can be used as general purpose digital outputs. In addition, any of these pins can be individually connected to a timer; T4 may be connected to pin P0, T5 to P1, T6 to P2 and T7 to P3. The structure of a Port P pin is shown in Figure 9-7.

The 16-bit PORTP Register is used both for holding the values presented on the pins and for determining each pin's function. While a pin is connected to a timer, the PORTP data bit becomes read-only to software, and is toggled whenever the corresponding timer underflows.



GL-14-0-U

Figure 9-7. Structure of Port P

The Port P pins cannot be set floating, and on reset they are initialized to the low state.

	1	Word at 0152														
PORTP	17 TFN 0	*	*	17 OUY 0	T6 TFN 0	*	٠	T6 OUT	15 TFN 0	*	*	TS OUT	I4 IFN 0	*	*	T4 OUT 0
	Pin P3 Control Pin P2 Control								Pin P1 Control Pin P0 Control							01
	Byte at 0153								Byte at 0152							

- * Unassigned bit, indeterminate when read.
- 0 Cleared at reset.
- X State on reset is indeterminate.

OUT Programs the initial Port P output signal level. These bits become read-only once Toggle mode has been entered by setting the TFN bit.

OUT = 1 Places the corresponding Port P pin at logic 1.

OUT = 0 Places the corresponding Port P pin at logic 0.

TFN Specifies the functions of the port pins PO(T4OUT)—P3(T7OUT).

TFN = 1 Enables the corresponding OUT bit to be toggled on an underflow from its Timer. While TFN is set, the OUT bit is read-only to software.

TFN = 0 Disables toggling. The Port P pin continues to take its value from the OUT bit, which may be changed by software.

NOTE: It is not recommended that an OUT bit be altered in the same access that sets the corresponding TFN bit to a one; doing so can affect the OUT bit unpredictably. Instead, set the OUT bit to the desired value first, then set the TFN bit to a one in a subsequent instruction. Clearing the TFN bit while changing the OUT bit in the same access is allowed, and performs both changes reliably.

_	
-	

BUS FUNCTIONS AND CONFIGURATIONS

10.1 BUS MODES AND SELECTION

The 16 pins of Port A and four of the Port B pins may be used as a bus for accessing off-chip memory and/or peripherals.

In this configuration, Port A becomes a 16-bit multiplexed address/data bus and the Port B pins B10, B11, B12 and B15 become, respectively, the bus control signals ALE, \overline{WR} , \overline{HBE} and \overline{RD} .

Three separate modes of operation must be selected to configure the HPC16083 to its application:

- Single Chip vs. Expanded Mode
- Normal vs. ROMless Mode
- 8-Bit vs. 16-Bit Mode

10.1.1 Single Chip Mode vs. Expanded Mode

The HPC16083 architecture recognizes a single 64-Kbyte addressing space (0000 to FFFF) containing all memory, registers and I/O addresses (memory-mapped). The on-chip addressing space consists of the first 512 bytes (addresses 0000 through 01FF), which contain the 256 bytes of RAM, registers and I/O, and the last 8 Kbytes (addresses E000 through FFFF), which contain the on-chip ROM. Two modes of operation are available which determine the allowable range of memory addressing. The Single Chip Mode (configuration shown in Figure 10-1) limits the addressing range to 8 Kbytes and enables a function of the Watchdog logic which detects illegal addresses outside of the 8K range. The Expanded Mode of operation enables memory accesses anywhere in the 64-Kbyte address range of the HPC with no illegal address detection of Watchdog (see Chapter 13). The on-chip RAM may always be accessed and code may even by placed in on-chip RAM and executed from there regardless of the mode selection.

The range of addresses used by the on-chip ROM are referred to as the "on-chip ROM range." This range is E000 to FFFF on the HPC16083. The ROM range may differ for different HPC family members (see appendices). The "on-chip ROM range" may physically be on-chip ROM or external memory, depending on whether the HPC16083 is operating in the normal or ROMless mode as determined by the state of the EXM pin, (see Section 10.1.2).

Single Chip Mode is selected by clearing the EA bit (bit 4 of the PSW register). The EA bit is initialized to "0" when the HPC comes out of reset so clearing the EA bit is not really necessary if the Single Chip Mode is desired. In Single Chip Mode, the HPC16083 is limited to memory accesses within the "on-chip ROM range" (E000 through FFFF) and the on-chip RAM and Register range (0000 through 01FF). Accesses outside of this range will generate a Watchdog Output (WO pin is pulled low) when the HPC16083 is in the Single Chip Mode. Depending on the application, this pin may be ignored, or it may be connected externally to an interrupt, the RESET pin or other circuitry in the user's system (see Chapter 13).

Expanded Mode is selected by setting the EA bit in the PSW register to "1." In Expanded Mode, memory accesses are permitted anywhere in the 64-Kbyte address range. The illegal address

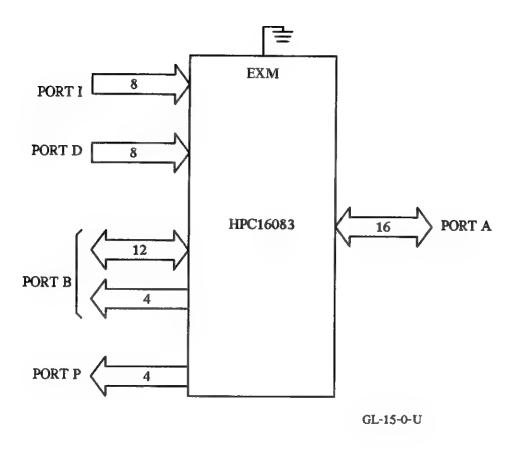


Figure 10-1. System Configuration: Single Chip Mode

detection feature of the Watchdog circuitry is disabled in Expanded Mode. All off-chip memory accesses cause activity on the external bus and control lines while the HPC is in the Expanded Mode. The user's program must set the EA bit in the PSW register to "1" causing the HPC to enter the Expanded Mode prior to accessing addresses between 0200 and DFFF. If this is not done, a Watchdog Output will occur and the bus will not function properly.

Because the HPC16083 enters the Single Chip Mode automatically at power-up and reset, the code executed on reset must be between E000 and FFFF. Once the EA bit is set to "1," the HPC16083 may then access memory outside of the on-chip ROM range and on-chip RAM and register range.

10.1.2 Normal Mode vs ROMless Mode

This mode setting determines whether the HPC16083 will use the on-chip 8-Kbyte ROM at addresses E000—FFFF (Normal Mode), or whether it will access these addresses using off-chip bus transfers (ROMless Mode).

ROMless mode provides a means of prototyping both Single Chip and Expanded mode applications. An external memory is used to replace the on-chip ROM. Since this requires use of the bus, Port A and the four bus control pins of Port B are not available for bit I/O directly from the HPC. These functions, however, can be re-created by external logic which responds to the appropriate addresses on the bus. (The PEARL chip, HPC16900, provides these functions in user applications; the MOLE system provides them in emulation environments.)

While ROMless Mode is enabled, all accesses to both on-chip and off-chip memory addresses cause bus transfers. Read accesses from on-chip RAM and most register locations (the exceptions are given in the note below) read data both from the on-chip source and from the bus, but accept the data from the on-chip source. (The value read from the on-chip source is not presented on the bus by the HPC.) Write accesses to on-chip destinations transfer data both to the on-chip destination and on the bus.

NOTE: There are three groups of on-chip registers which are treated as exceptions in ROMless mode:

- Addresses 00E0--00FF, which are the addresses of the control registers for Ports A and B. These registers are handled as special cases in order to allow external re-creation of the functions of those pins that are claimed for the bus. Of these registers, PORTA, DIRA and UPIC are mapped entirely off-chip in ROMless Mode, meaning that accesses to these locations cause data to be transferred on the bus instead of accessing the on-chip location. The PORTB, DIRB and BFUN registers accept all data written to them, but that data is also sent out on the bus where it can be captured by external re-creation logic to replicate the functions of pins B10, B11, B12 and B15, which are claimed for bus control functions.
- Addresses 0160—017F, which are addresses reserved for on-chip custom functions. In ROMless mode, these addresses are all mapped off-chip to allow prototyping. In other modes, these addresses are not connected to any on-chip hardware and do not cause external bus transfers. This feature is specific to the HPC versions without custom hardware in these addresses. Versions of the HPC which incorporate custom hardware do not map it off-chip, regardless of the mode.
- The dedicated registers A, B, X, K, SP and PC, although they are memory-mapped, are not actual memory locations. They are usually loaded or altered internally, without memory-like accesses visible from the bus. For example, an instruction that loads the A register from off-chip memory will read the value from the bus directly into the A register, without performing a separate write cycle.

ROMless Mode is entered whenever the input pin EXM is set high. While EXM is low, the HPC is in Normal Mode.

NOTE: The EXM signal is neither sampled internally nor in any other way synchronized to internal chip activity. It should be brought to its correct state during the RESET pulse and left there. It is not recommended that the state of EXM be altered, while a program is running, for any purpose other than chip testing.

10.1.3 8-Bit vs. 16-Bit Mode

In all modes above which use Port A as a bus, a choice must also be made as to the size of the data bus.

In 16-Bit Mode, all 16 pins of Port A are used to transfer data as well as addresses. Two octal latches are required externally to hold each address as it is issued by the HPC; the signal ALE from the HPC clocks the latches. See Figure 10-2.

In 8-Bit Mode, only pins A0-A7 are used to transfer data. Pins A8-A15 continue to hold the most-significant eight bits of the address; they need not be latched externally. This is in

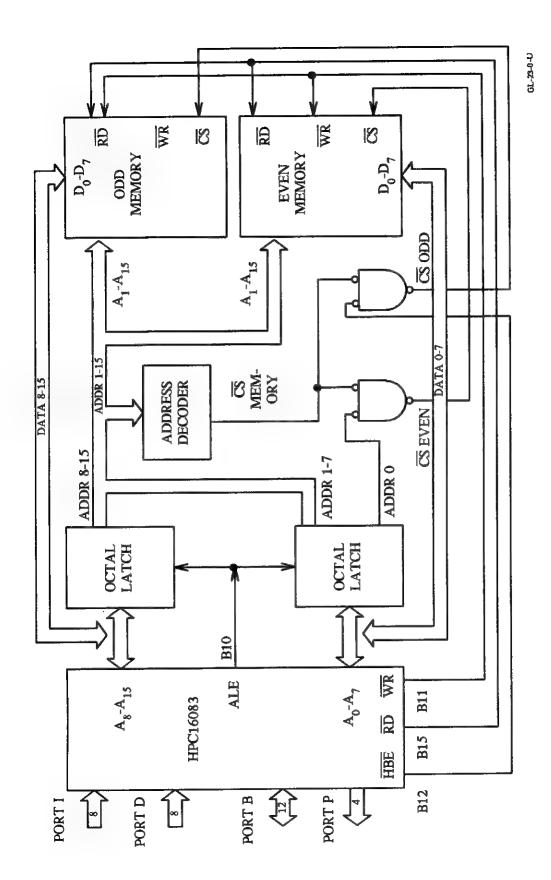


Figure 10-2. System Configuration: 16-Bit Modes, Gating Chip Selects

general a more cost-effective mode of operation, as only a single octal latch is required to hold the multiplexed portion of the address, and the external system bus and memory is only eight bits wide (see Figure 10-3).

In 8-Bit Mode, data on the external bus must be accessed only as 8-bit values. This also applies to any registers being re-created with off-chip hardware in ROMless Mode. Attempting to perform a 16-bit access will actually transfer only eight bits of the value. The programming tradeoffs involved are discussed in Section 10.9. Data in the on-chip ROM may be accessed as 8-bit or 16-bit values regardless of the external bus mode.

The bus size is determined once, at reset, by the state of the pin B12 (HBE). HBE is sampled by the HPC on the rising edge of the RESET pulse. If it is sampled high, the HPC enters 8-Bit Mode. If it is sampled low, the HPC enters 16-Bit Mode. A weak internal pull-down device is activated during the RESET pulse, so that if no connection is made to HBE the HPC defaults to 16-Bit Mode.

NOTE: Accessing 16-bit values at odd byte addresses is never legal in the HPC family, regardless of the mode, and the result, if attempted, is not generally useful. When the user writes a 16-bit value to an odd address location, only the upper byte of the 16-bit value is written into the addressed location. No other memory locations are affected. When the user reads a 16-bit value from an odd address location, this results in a 16-bit value whose upper byte contains the data from the addressed location. The lower byte of the 16-bit value read back will contain undefined data.

10.2 SYSTEM CONFIGURATION TABLES

Table 10-1 shows the modes selected by the EXM pin and the PSW EA bit.

Table 10-2 lists the various memory areas within the HPC addressing space, and how their use differs with the bus mode selected.

10.3 BUS OPERATION

In its use of the bus, the HPC enters four types of bus state:

- TA (Address), during which it emits an address and internal status,
- TD (Data), during which data is transferred,
- TW (Wait), during which it waits for a transfer to be completed,
- TI (Idle), during which the bus is idle between transfers.

The bus changes its state on each rising edge of the CK2 clock.

Figures 10-4 and 10-5 show the simplest cases: Read and Write accesses performed with no Wait states. (At least one Wait state is often mandatory, however. See Section 10.4.)

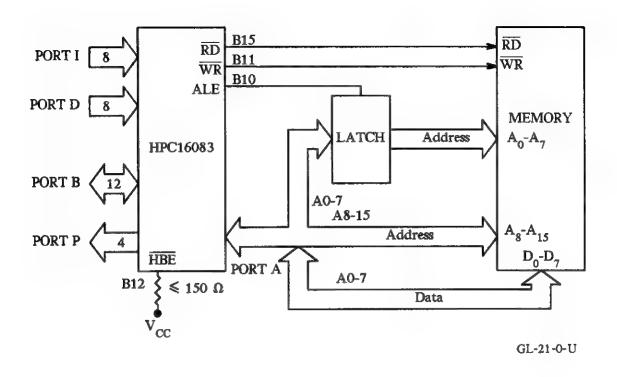


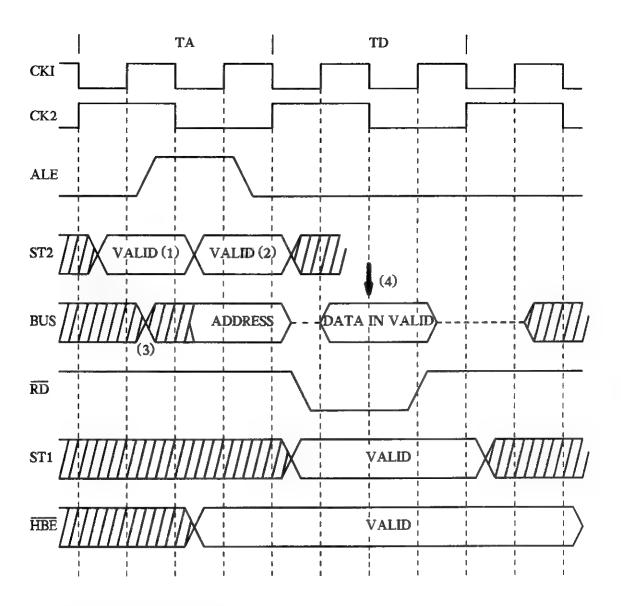
Figure 10-3. System Configuration: 8-Bit Modes

TABLE 10-1. SYSTEM CONFIGURATION CHART

EXTERNAL MEMORY PIN EXM	EXTERNAL ACCESS PSW BIT 4	OPERATION MODE
0	0	Single Chip Normal
0	1	Expanded Normal
1	0	Single Chip ROMless
1	1	Expanded ROMless

TABLE 10-2. MEMORY SPACES AND ACCESSIBILITY

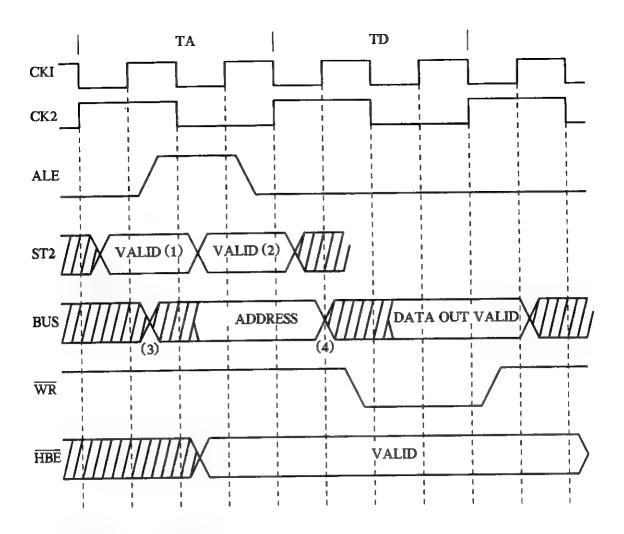
ADDRESS SPACE	SC NORMAL MODE	EXPANDED NORMAL MODE	ROMless MODE	MEMORY AREA	
0000:00BF 00C0:00DF 00E0:00FF	on on on†	on on on*†	on on off**†	On-Chip RAM Core Registers On-Chip Peripheral Registers: Port A, Port B and UPI Interface On-Chip Peripheral Registers	
0160:017F 0180:019F 01A0:01BF 01C0:01FF 0200:DFFF	on***† on X on X†	on***† on X on off†	off† on X on off†	(Provision for Custom Logic) On-Chip Peripheral Registers On-Chip RAM Off-Chip Space	
E000:FFFF	on†	ont	off†	(Causes Watchdog error in Single-Chip Mode.) ROM Space	
on off X	=	On-chip available memory Off-chip available memory Unavailable memory			
*		All registers inaccessible except PORTB, DIRB and BFUN.			
**		All registers except PORTB, DIRB and BFUN. Bits 10, 11, 12 and 15 of PORTB, DIRB and BFUN must be re-created off-chip.			
alcolor.		These addresses reference on-chip ROM locations F160—F17F in these modes.			
*		Wait states are recognized in transfers within this range. In 8-bit modes, these areas must be accessed only as bytes; in particular, the stack must not reside here. If bits 10, 11, 12 and 15 of PORTB, DIRB and BFUN are not being used, these registers can still be accessed as 16-bit values.			



- (1) HPC presents first ST2 value.
- (2) HPC presents second ST2 value.
- (3) HPC starts driving bus to present address.
- (4) HPC samples input data.

GL-16-0-U

Figure 10-4. Bus Timing: Read Cycle; No Wait States



- (1) HPC presents first ST2 value.
- (2) HPC presents second ST2 value.
- (3) HPC starts driving bus to present address.
- (4) HPC starts driving bus to present data.

GL-17-0-U

Figure 10-5. Bus Timing: Write Cycle; No Wait States

During TA, the address strobe ALE (Address Latch Enable) is pulsed high. On the rising edge, the ST2 pin becomes valid, holding one bit of the processor's internal status code (see Section 10.7). On the falling edge, the second bit of internal status is presented on ST2 and the address is valid on Port A for the address latch. (See Figure 10-2 for a typical system connection diagram.) In 16-Bit Mode (Section 10.1.3), the signal HBE (High Byte Enable, Section 10.6) becomes valid with the address.

During TD, data is transferred:

In a Read transfer (Figure 10-4), the \overline{RD} strobe goes active and the Port A pins switch to inputs. Upon reading the data presented on the bus by the external device, the HPC removes the \overline{RD} strobe. The status signal ST1 goes high during TD, as shown, if the first byte of an instruction is being fetched.

In a Write transfer (Figure 10-5), data is presented on the Port A pins and the \overline{WR} signal is pulsed. Note that the rising edge of \overline{WR} and the removal of valid data actually occur in the bus state following TD. This following bus state can be either TI or the TA state of another transfer.

TI is an idle bus state, during which the HPC is performing internal computations and no bus transfer is in progress. During TI states the ALE, \overline{RD} and \overline{WR} strobes are inactive. The bus pins (Port A) may be driven by the HPC, if it writes to an on-chip location during the TI state.

TW is a "Wait state," used to extend bus transfers for slower memories and peripherals, or onchip ROM above certain speeds. See Section 10.4.

TA states are not always followed immediately by TD states, as the HPC will at times pulse ALE to report its internal status and/or to issue an address which might be used later. Nor is it always the case that a TD state will be immediately preceded by a TA state:

- Read/Modify/Write operations are performed using the sequence: TA, TD(read),
 TD(write). This sequence appears in accessing the destination operand of the instructions
 ADC, ADD, AND, DADC, DIV, DIVD, MULT, OR, XOR, DECSZ, INC, X, XS and the Set
 and Reset bit instructions. Some TI states may appear between the TD states, depending
 on the instruction.
- In the instructions SUBC, DSUBC, MULT, DIV, DIVD, and IFGT, another type of access is performed, in which the destination operand is read twice before being used. The sequences used are:

TA, TD (read), TI, TD (read), TD (write) for the SUBC instruction,

TA, TD (read), TI, TD (read), TI, TD (write) for the DSUBC instruction,

TA, TD (read), TD (read), many TI's, TD (write) for the MULT, DIV, and DIVD instructions,

and TA, TD (read), TI, TD (read) for the IFGT instruction.

10.4 WAIT STATES

To support accesses to slower peripheral devices and memories, the HPC allows extension of bus accesses by insertion of "Wait" states (TW). Wait state insertion lengthens the \overline{RD} or \overline{WR} strobe in units of the CK2 clock period, and relaxes the access time requirement by the same amount.

Insertion of at least one Wait state is often mandatory, as well, for the following reasons:

- The on-chip ROM may require one Wait state for accesses at higher clock frequencies. See the data sheet for details.
- External hardware cannot request Wait states unless at least one Wait state is preprogrammed into transfers (using the WAITn bits in the PSW register; see below).
- At higher crystal frequencies, it is not guaranteed that an off-chip Read cycle without
 Wait states can input data reliably, regardless of memory access time. This is because the
 HPC samples data from the bus at a time when it may still be asserting the address from
 a previous TA state. See the data sheet for details.

Operation with no Wait states, then, should be selected only at lower clock speeds, and only in applications where no Wait states are ever needed.

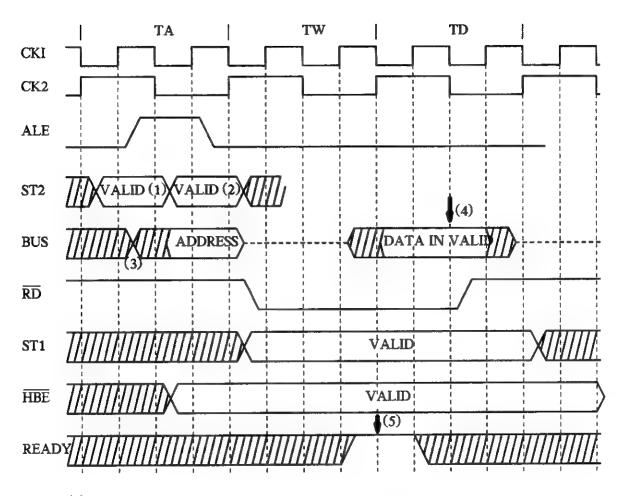
Figures 10-6 and 10-7 show typical Read and Write transfers with one Wait state (TW) inserted. During the first TW, the appropriate strobe (RD or WR) goes active. The Ready signal (Section 10.5) is sampled during each Wait state to determine whether more Wait states are to be added. During TD, data is transferred and the strobes are removed.

Wait states are inserted (when requested) in accesses to off-chip memory, the on-chip ROM, and to the on-chip I/O registers in the address ranges 00E0—00FF and 0160—017F, because they are mapped off-chip in ROMless modes (see Section 10.1.2). Accesses to the other on-chip registers and to the on-chip RAM, however, never incorporate any Wait states.

10.5 REQUESTING WAIT STATES

There are two methods by which Wait states can be requested:

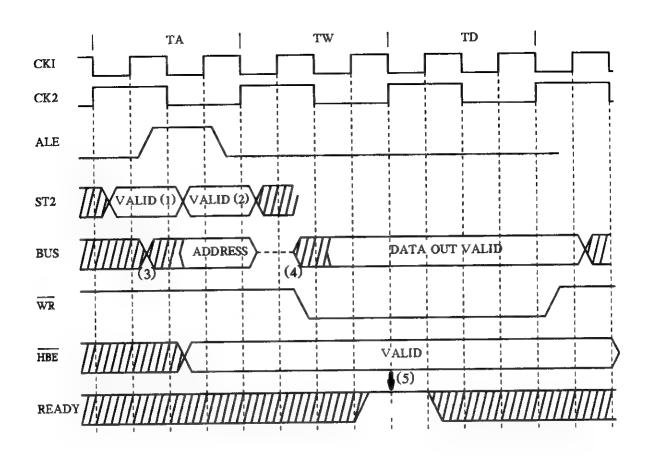
- By setting the WAIT1/WAIT0 bits in the PSW register appropriately.
- By pulling the RDY/HLD pin low. This pin must have been configured for the Ready function, and at least one Wait state must have been requested in the PSW register, as described below.



- (1) HPC presents first ST2 value.
- (2) HPC presents second ST2 value.
- (3) HPC starts driving bus to present address.
- (4) HPC samples input data.
- (5) HPC samples READY signal.

GL-18-0-U

Figure 10-6. Bus Timing: Read Cycle; One Wait State



HPC presents first ST2 value.
 HPC presents second ST2 value.
 HPC starts driving bus to present address.
 HPC starts driving bus to present data.
 HPC samples READY signal.

GL-19-0-U

Figure 10-7. Bus Timing: Write Cycle; One Wait State

The bits WAITO and WAIT1 in the PSW register (bits 2 and 3, Section 8.5) allow 0, 1, 2 or 4 Wait states to be pre-programmed into all applicable transfers. The number of Wait states selected in this manner is a minimum, and can be increased by external hardware using the Ready feature (below). The settings of the WAITO and WAIT1 bits are defined as follows:

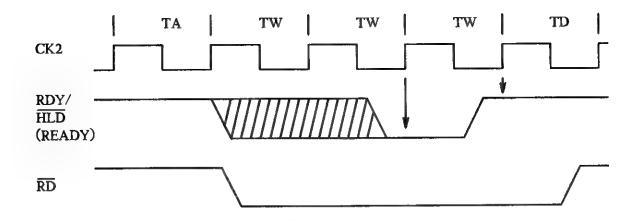
Bit WAIT1	Bit WAITO	Wait States
0	0	4
0	1	2
1	0	1
1	1	0

On reset these bits are cleared, selecting four Wait states as the default.

The number of Wait states selected in the PSW register can be extended by use of a Ready signal, provided to the HPC on the RDY/HLD pin. This pin has another function by default (the HOLD function, Section 11.1), and therefore must be configured for the Ready function first. This is done by setting the control bit RDY/HLD in the Interrupt Condition register IRCD (Section 15.5) to a "1".

The Ready input indicates, when high, that the external peripheral or memory device is ready to complete the data transfer in progress. If Ready is sampled low at the point indicated in Figures 10-6 and 10-7, the bus cycle is extended by one more Wait state. During this additional Wait state, the Ready signal is sampled again. This continues until the Ready input is sampled high, after which the TD state is entered and the transfer is completed.

When more than one pre-programmed Wait state is requested, the Ready signal is ignored until the last pre-programmed Wait state is entered. The Ready signal is then sampled, and serves to extend the bus cycle further at that point. Figure 10-8 shows the proper sequence and the result of requesting one additional Wait state with the Ready signal in a system with two pre-programmed Wait states.



USE OF READY SIGNAL WITH TWO PRE-PROGRAMMED WAIT STATES (READ TRANSFER SHOWN)

FIRST TWO TW STATES ARE PRE-PROGRAMMED.
ARROWS INDICATE POINTS WHERE READY SIGNAL IS SAMPLED.

GL-20-0-U

Figure 10-8. READY Signal Timing

- NOTES: 1. The RDY/HLD pin is held high with an on-chip pull-up device (requesting no Wait states) if no connection is made to it.
 - 2. The Ready signal should not be held low except while the RD or WR strobe is active. This is because some on-chip transfers, which do not pulse RD or WR, can still be susceptible to Wait states. For example, instruction fetches from the on-chip ROM in Expanded Normal Mode or Single-Chip Normal Mode are affected by the Ready signal (even though they are not seen externally), as long as at least one Wait state is pre-programmed in the PSW register. (Note that four pre-programmed Wait states are requested by default on Reset.)
 - 3. The Ready signal must remain stable throughout the setup and hold periods given in the data sheet to avoid possible metastable states within the HPC. Unpredictable system operation can result if Ready is applied asynchronously with respect to the points at which it is sampled.
 - 4. In the ROMless modes of operation, transfers performed to and from the RAM and most internal registers are seen on the bus, as explained in Section 10.1.2 Most of these transfers have no Wait states, regardless of the PSW WAITn bits and the Ready signal, but it is also not necessary for external memory to provide data in such transfers. They are

performed for purposes of emulation and tracing only. Any transfer that might require reading from off-chip memory, even for emulation purposes, contains the pre-programmed number of Wait states and responds to the Ready signal. See Section 10.2 for further details.

10.6 ADDRESSING BYTE/WORD EXTERNAL MEMORIES

When the HPC is operating in 16-Bit Mode, each bus-width unit of data contains two bytes with separate addresses (one even, one odd). Either or both of these bytes can be accessed by the HPC in a bus transfer.

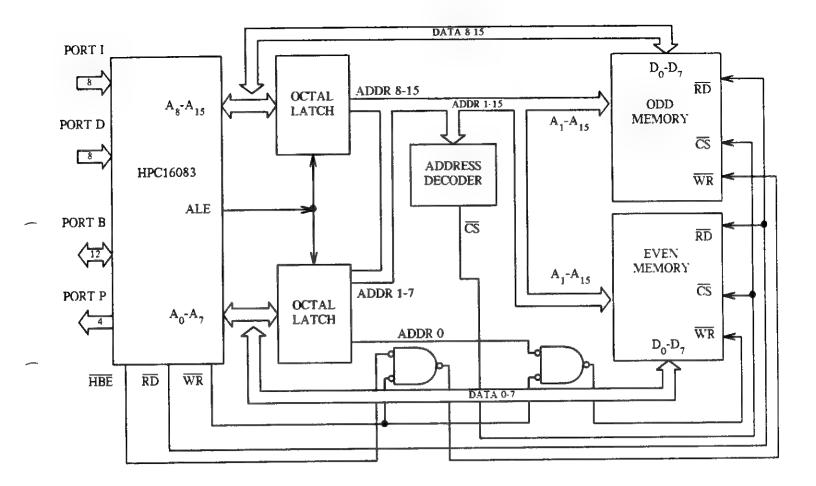
To support such accesses, memory for a 16-bit HPC system is divided into two eight-bit banks, as shown in Figure 10-2: an "even" bank, which contains all bytes whose addresses are even, and an "odd" bank, which contains all odd-addressed bytes. The odd bank is connected to the high-order half of the bus (A8—A15) and the even bank is connected to the low-order half (A0—A7).

Because consecutive bytes in each bank of memory differ in address by two rather than one, the least-significant bit of address going to memory is address bit A1 rather than A0. The selection between the two bytes on the bus is made by using address bit A0 and another signal ($\overline{\text{HBE}}$) to enable access to one or both of those bytes. $\overline{\text{HBE}}$ is a low-active signal signifying an access to the high-order byte on the bus (the odd-addressed byte). It becomes active when the HPC is accessing either an odd-addressed byte or an even-addressed word. (Odd-addressed word accesses are illegal; see Section 10.9.) Address bit A0, when low, indicates that the HPC is accessing at least the low-order byte of the bus (the even-addressed byte). The functions of $\overline{\text{HBE}}$ and A0 are summarized in the table below.

HBE	AO	ACCESS
0 0 1 1	0 1 0 1	Word Access High-Order (Odd) Byte Access Low-Order (Even) Byte Access (Not Used)

A simple system design is shown in Figure 10-9. The signals \overline{HBE} and A0 are used to gate the \overline{WR} strobe into an Odd Write Strobe and an Even Write Strobe. No additional logic is required on the \overline{RD} strobe; for a Read transfer the HPC floats the entire bus and ignores any unneeded data placed there. Care should be taken to ensure that the additional propagation delay placed on the gated Write strobes does not cause a data hold time violation (See the data sheet for the relevant values.) Note that if all of the external memory is read-only, the signals A0 and \overline{HBE} (as well as \overline{WR}) are not needed.

Another approach, which does not delay the Write strobe, is shown in Figure 10-2. Here \overline{HBE} and A0 are used to gate the address decoding logic, providing separate Chip Select signals to the odd and even banks of memory. A similar but potentially faster approach is to use two separate address decoders, one enabled while \overline{HBE} is low and the other enabled while A0 is low.



GL 22-0 t

Figure 10-9. System Configuration: 16-Bit Modes, Gating Write Strobe

NOTE: Instruction fetches, which are always one-byte read transfers, often do not have \overline{HBE} set according to these rules. An instruction fetch from an even address (A0 = 0) may have \overline{HBE} active, thus appearing to be a 16-bit read. \overline{HBE} is always active, of course, when an instruction byte is fetched from an odd address (A0 = 1).

The HBE pin also serves as a strap input, sampled on the rising edge of RESET, selecting 8-Bit Mode or 16-Bit Mode. See Section 10.1.3.

10.7 STATUS PINS ST1 AND ST2

The status pins ST1 and ST2 provide, for emulation and testing purposes, some additional information regarding the internal state of the HPC. They are internally synchronized to bus activity.

The ST1 pin is normally low, and goes high through the TW and TD states of a Read transfer when the HPC is reading the first byte of an instruction. See Figure 10-4. The ST1 signal goes high not only on the first byte of an instruction, but also on reading the Opcode byte (which will be the last byte rather than the first when the instruction has an addressing directive prefix). This information is helpful in disassembling instructions from a trace of bus activity.

The ST2 signal, in conjunction with the two edges of the ALE signal, provides the following information (see Figures 10-4 and 10-5):

VALUE OF ST2 ON RISING EDGE OF ALE	VALUE OF ST2 ON FALLING EDGE OF ALE	INTERNAL ACTIVITY
0	0	Normal; no special activity.
0	1	Interrupt service is starting.
1	o	An instruction is being skipped.
1	1	A new instruction is starting execution.

Because the HPC prefetches instructions before executing them, the ST1 and ST2 pins provide different information. ST1 indicates when the first byte of an instruction is fetched from memory. ST2 indicates when the instruction actually begins execution, or why it will not be executed.

NOTE: At higher crystal frequencies, it is not guaranteed that the edges of the ALE pulse can be used directly to latch the values presented on the ST2 pin. See the data sheet.

10.8 SYSTEM BUS BEHAVIOR

The HPC System Bus consists of the 16 pins of Port A and four pins from Port B. Port A is used as the Address/Data bus and the four pins B10, B11, B12 and B15 of Port B are used as the bus control functions ALE, WR, HBE and RD respectively.

On reset both Port A and Port B are set floating. If the EXM pin is held high (selecting ROMless mode), the two pins \overline{WR} (B11) and \overline{RD} (B15) are driven to their inactive states immediately after the \overline{RESET} pulse ends. If the pin \overline{HBE} (B12) is also left open or is held low (selecting a 16-bit ROMless mode), the HPC begins driving it immediately after the \overline{RESET} pulse ends. Depending on the exact timing of the rising edge of the \overline{RESET} pulse, the ALE signal (on B10) may pulse once or twice after reset while the HPC issues an indeterminate address, but a valid address is guaranteed to be issued before the first data strobe (\overline{RD} or \overline{WR}) is pulsed.

For all modes except Single Chip Normal Mode the HPC uses the bus to interface with external memory. When interfacing with external memory, the on-chip PORTA data register is dedicated to holding the address/data values.

In Expanded Normal Mode, the I/O registers for Port A are not available to the user. A write to the PORTA data register will momentarily load the PORTA data latches with the data written to the port. This data, however, will be immediately overwritten by the address of the next memory location accessed. Attempting to read the PORTA data latches will return the address of the PORTA register. The HPC also forces the on-chip DIRA register, the direction register for Port A, to contain all zeroes. All further attempts to modify DIRA are disabled.

See Table 10-3 for a summary of the memory organization in the various operating modes.

Bus behavior is defined below for the seven possible cases:

Single Chip Normal	Section 10.8.1
Expanded Normal 8-bit	Section 10.8.2
Expanded Normal 16-bit	Section 10.8.3
Single Chip ROMless 8-bit	Section 10.8.4
Single Chip ROMless 16-bit	Section 10.8.5
Expanded ROMless 8-bit	Section 10.8.6
Expanded ROMless 16-bit	Section 10.8.7

10.8.1 Single Chip Normal Mode

In Single Chip Normal mode there is no off-chip bus. Port A is an I/O port and the four bus control pins of Port B are bit-programmable outputs. (They cannot be used as input pins, but designating them as such in the DIRB register will cause them to float.) The \overline{HBE} strap input setting is ignored while the HPC is in this mode; hence all internal RAM, ROM and register accesses may be performed 8 bits or 16 bits at a time. Attempted access to an off-chip address causes a Watchdog event, as described in Chapter 13.

TABLE 10-3, MEMORY ORGANIZATION/AVAILABILITY CHART

EXM PIN	EA BIT (PSW)	8/16 HBE PIN	MODE	INTERNAL ROM	RD SIGNAL	WR SIGNAL	INTERNAL REGS. & RAM	ROM/BUS DATA RESTRICTION
0	0	x	Single Chip Normal	YES	N/A	N/A	16	None
0	1	0	Expanded Normal 16 Bit	YES	@	@	16	None
0	1	1	Expanded Normal 8 Bit	YES	@	@	16 *	8*
1	0	0	Single Chip ROMless 16 Bit	NO	YES	YES	16	None
1	0	1	Single Chip ROMless 8 Bit	NO	YES	YES	16 *	8 *
1	1	0	Expanded ROMless 16 Bit	NO	YES	YES	16	None
1	1	1	Expanded ROMless 8 Bit	NO	YES	YES	16 *	8*

x Don't Care.

Registers in address ranges 00E0—00FF and 0160—017F, if accessible, must be accessed one byte at a time. (Registers PORTB, DIRB and BFUN may still be accessed 16 bits at a time if bits 10, 11, 12 and 15 are not being used in the application.)

The stack must reside in on-chip RAM.

© Control signals RD and WR pulsed for off-chip address locations only (200 - DFFF). ALE is pulsed for all addresses.

^{*} The HPC16083 ROM area (Address range X'E000 to X'FFFF) cannot be accessed as 16-bit data.

10.8.2 Expanded Normal 8-bit

In this mode the total 64K address space of the HPC is accessible. This space includes the on-chip ROM, RAM and registers. Off-chip memory is accessed with an 8-bit data bus. Port B pin B12 ($\overline{\rm HBE}$) must be tied to $V_{\rm CC}$. Data in the on-chip ROM may be accessed as bytes or as 16-bit words regardless of 8-bit mode. Programming restrictions apply in accesses to off-chip memory. See Section 10.9.

Immediately upon entering this mode Port A becomes the Address/Data bus. Port B pins B15, B11 and B10 become the RD, WR and ALE signals, respectively.

In Expanded Normal mode, all addresses, both off-chip and on-chip, are issued, and the ALE signal is pulsed for each.

Read and write operations are performed as follows:

Read: The RD signal is generated only if the addressed location is off-chip. Only the

low-order eight bits of the bus are set floating to allow input. The high-order eight bits continue to hold the most-significant byte of the address.

eight one continue to hold the most significant of a or the address

Data being read back from on-chip addresses is not presented on the bus.

Write: The WR signal is generated only if the addressed location is off-chip. Only the low-order eight bits of the bus hold the value to be written. The high-order

eight bits continue to hold the most-significant byte of the address.

Data being written to all locations, including the on-chip addresses, is also presented on the low-order eight bits of the data bus. Only the least-significant eight bits of 16-bit data written internally are visible, however.

10.8.3 Expanded Normal 16-bit

In this mode the total 64K address space of the HPC is accessible. This space includes the onchip ROM, RAM and registers. External memory is accessed with the data bus defined to be 16 bits wide.

Immediately upon entering this mode Port A becomes the bus. All four bits of the Control Bus (RD, WR, ALE and HBE) are used.

In Expanded Normal mode all addresses, both off-chip and on-chip, are issued, and the ALE signal is pulsed.

Read and write operations are performed as follows:

Read: The RD signal is generated only if the addressed location is an off-chip

location. All sixteen bits of the Address/Data bus are set floating to accept data.

Data being read from on-chip addresses is not presented on the data bus.

Write: The WR signal is generated only if the addressed location is an external location. A word value is presented on all 16 bits of the bus. A byte value is

presented on both the high-order and low-order halves of the bus.

Data being written to all locations, including the on-chip addresses, is

presented on the data bus.

10.8.4 Single Chip ROMless 8-bit

This mode, unlike other ROMless modes, is not properly a prototyping mode, since there is no corresponding "Single Chip Normal 8-Bit" mode. It can be useful, however, in providing a low-cost system solution for small applications which do not lend themselves to use of an on-chip mask-programmed ROM. (Eventual migration into the on-chip ROM is possible, however, without any change to the program. It would simply not take full advantage of the 16-bit capabilities of the HPC in Single Chip Mode.) As shown in Figure 10-3, such a system requires only a single octal latch (74x373 type) and a ROM or EPROM externally, since the HPC holds the eight high-order address bits on the upper half of Port A.

The HPC16083 is restricted to accessing only the single-chip ROM range (E000-FFFF) and on-chip RAM and register range (0000-01FF) addresses. In Single Chip ROMless 8-bit mode (only), accesses to logically off-chip addresses (0200—DFFF) do not transfer correct data on the bus, although the strobe timing and functions are correct. The PSW EA bit should always be turned on (changing the mode to Expanded) if such accesses are to be performed. The $\overline{\rm WO}$ pin (Watchdog Output) is pulsed low in all Single Chip modes if the program accesses a logically off-chip memory address. External memory may only be accessed as bytes, see Section 10.9.

Immediately upon entering this mode Port A becomes the Address/Data bus. The HPC operates with the data bus defined to be eight bits wide. Port B pin B12 ($\overline{\text{HBE}}$) must be tied to V_{CC} . In ROMless modes all addresses, both off-chip and on-chip, are issued.

Read and Write operations are performed as follows:

Read:

The \overline{RD} signal is generated for all Read accesses, both off-chip and on-chip. The HPC, however, only accepts data (from the lower 8 bits of Port A) when it is accessing a memory location mapped onto the on-chip ROM space (E000—FFFF), or a register within the ranges 00E0—00FF and 0160—017F (see Section 10.1.2). Because of the eight-bit bus, locations in these ranges must be accessed as individual bytes in this mode; never as 16-bit words. (The Port B control registers PORTB, DIRB and BFUN may still be accessed as 16-bit registers if bits 10, 11, 12 and 15 are not being used in the application, see Section 10.9.)

The low-order byte of the bus floats for a read operation. The upper byte of the bus continues to hold the address. Data being read back from on-chip locations is not seen on the system bus.

Write:

The WR signal is generated in writing to all addresses, both on-chip and off-chip. Data being written to both on-chip and off-chip destinations is presented on the low-order byte of Port A. 16-bit writes to on-chip registers and RAM display only the low-order eight bits on the bus.

10.8.5 Single Chip ROMless 16-bit

This mode is provided to allow prototyping of Single Chip Mode systems. In this mode, the HPC16083 is restricted to accessing only the logical on-chip memory space. The WO pin (Watchdog Output) triggers if the part attempts to access a logical off-chip memory location (0200—DFFF).

Immediately upon entering this mode, Port A becomes the Address/Data bus. In this mode, the part operates with the data bus defined to be sixteen bits wide. Addresses are presented for all accesses, both on-chip and off-chip. Port B pins (RD, WR, ALE, and HBE) are used for control.

Read and Write cycles are performed as follows:

Read:

The RD signal is generated for all addresses, both on-chip and off-chip. The HPC16083, however, only accepts data when it is accessing a memory location mapped onto the on-chip ROM space (E000—FFFF), or a register within the ranges 00E0—00FF and 0160—017F (see Section 10.1.2). All sixteen bits of the Address/Data bus are floated during a read operation.

Data being read back from on-chip registers and RAM locations is not seen on the system bus.

Write:

The \overline{WR} signal is generated for all addresses, both on-chip and off-chip. When writing a word, the data bus holds all sixteen bits of data to be written. When writing a single byte, that same byte value is presented on both halves of the bus.

10.8.6 Expanded ROMless 8-bit

In this mode Port A becomes the Address/Data bus. Port B pins B10, B11 and B15 (ALE, \overline{WR} and \overline{RD} , respectively) are used for bus control. Pin B12 (\overline{HBE}) is not used for bus control, and must be tied to V_{CC} . Addresses are presented for all accesses, both on-chip and off-chip. 8-bit programming restrictions apply, see Section 10.9.

Read and Write operations are performed as follows:

Read:

The $\overline{\text{RD}}$ signal is generated for all Read accesses, both off-chip and on-chip. The HPC16083, however, only accepts data (from the lower 8 bits of Port A) when it is accessing a memory location mapped onto the on-chip ROM space (E000—FFFF), the off-chip space (0200—DFFF) or a register within the ranges 00E0—00FF and 0160—017F (see Section 10.1.2). Because of the eight-bit bus, locations in these ranges must be accessed as individual bytes in this mode; never as 16-bit words. (The Port B control registers PORTB, DIRB and BFUN may still be accessed as 16-bit registers if bits 10, 11, 12 and 15 are not used in the application, see Section 10.9.)

The low-order byte of the bus floats for a read operation. The upper byte of the bus continues to hold the address. Data being read back from on-chip locations is not seen on the system bus.

Write:

The WR signal is generated in writing to all addresses, both on-chip and off-chip. Data being written to both on-chip and off-chip destinations is presented on the low-order byte of Port A. The upper byte of Port A continues to hold

the address. 16-bit writes to on-chip registers and RAM display only the low-order eight bits on the bus.

10.8.7 Expanded ROMless 16-bit

In this mode Port A becomes the Address/Data bus. All four bus control pins of Port B (\overline{RD} , \overline{WR} , \overline{HBE} and ALE) are used. Addresses are presented for all accesses, both on-chip and off-chip.

Read and Write operations are performed as follows.

Read:

The $\overline{\text{RD}}$ signal is generated for all Read accesses, both off-chip and on-chip. The HPC16083, however, only accepts data when it is accessing a memory location mapped onto the on-chip ROM space (E000—FFFF), the off-chip space (0200—DFFF) or a register within the ranges 00E0—00FF and 0160—017F (see Section 10.1.2).

All sixteen bits of the bus float for a read operation. Data being read back from on-chip locations is not seen on the system bus.

Write:

The WR signal is generated in writing to all addresses, both on-chip and offchip. All sixteen bits of the Address/Data bus hold the data to be written. If a byte is to be written, both the upper and lower halves of the bus hold the same byte data to be written.

10.9 PROGRAMMING CONSIDERATIONS IN 8-BIT MODES

The following hardware restrictions apply when using the HPC in an 8-Bit mode:

- All accesses to off-chip locations (i.e., all accesses using the bus) are restricted to being one-byte transfers.
- The on-chip registers in the address ranges 00E0—00FF and 0160—017F must also be
 accessed only as bytes. This is to allow re-creation of these register functions off-chip in
 ROMless mode using the 8-bit bus. The registers PORTB, DIRB and BFUN are exceptions;
 they may be accessed as 16-bit words, but only if bits 10, 11, 12 and 15 are not used.

All on-chip locations (RAM and registers) may be accessed as bytes or as words, without restriction.

The hardware restrictions above lead to the following programming considerations when using the HPC in an 8-Bit mode:

 No instruction may transfer a 16-bit value within the addressing range 0200—FFFF (offchip memory). For example, the following instruction:

LD A, W(X'F038)

will not work correctly, if X'F038 is an off-chip address.

The JIDW instruction becomes illegal because it insists on accessing its jump table entries
as words. The same effect, however, can be obtained by using JID to jump to one of a set
of JMP or JMPL instructions.

• All instructions that push or pop information on the stack also insist on performing 16-bit reads and writes. In 8-Bit modes, therefore, the stack must be held within on-chip RAM, which is the only 16-bit read/write memory present.

All other accesses to ROM are automatically one-byte accesses, and will work in any mode. In particular:

- Instructions are always fetched one byte at a time. 16-bit immediate values and addressing fields are assembled internally, and are not restricted.
- Automatic references to the Interrupt Vector Table and the JSRP (Subroutine) Vector Table are also performed as 8-bit transfers, and cause no problems.

SHARED MEMORY SUPPORT (DMA)

11.1 INTRODUCTION

The HPC16083 provides support for shared external memory access and DMA by means of two pins. These two pins are RDY/HLD, which receives the bus request HOLD, and B7, which outputs the Hold Acknowledge signal HLDA. RDY/HLD is an input and HLDA is an output. Figure 11-1 shows a system in which two HPC's share a block of memory.

HOLD is a low-active input to the HPC, on which an external device requests access to the system bus. The HPC responds to the HOLD request at the beginning of its next access (on-chip or off-chip). It acknowledges the HOLD request by setting the HLDA output low and floating the bus (Port A). Instruction processing is suspended until the HOLD request is removed.

11.2 TIMING SEQUENCES

The sequence of events in processing a HOLD request is shown in Figure 11-2. The RDY/HLD pin is sampled at the beginning of each TA bus state (Section 10.3), on the rising edge of the CK2 clock. Note that TA states are generated for on-chip accesses as well as external bus accesses. If HOLD is sampled low, the Port A (bus) pins float instead of presenting an address. At the next CK2 rising edge, the HLDA pin goes low to acknowledge the request, and the HPC enters internal Hold states (TH). The bus control signals ALE, RD, WR and HBE do not float; ALE is pulsed high during each TH cycle, RD and WR are held in their inactive states (high), and HBE (in 16-bit modes only) assumes the appropriate state for the transfer that the HPC is waiting to perform next.

The process of exiting from Hold states is shown in Figure 11-3. When the $\overline{\text{HOLD}}$ request is removed, the HPC removes $\overline{\text{HLDA}}$ in the next TH state. The HPC then reclaims control of the bus in a sequence of TH states, as shown, and performs a TA cycle to issue its next address and bus status.

- NOTES: 1. The HPC waits until the first access after the HOLD request to respond to it. For instructions performing multiplication or division, then, the delay before the response can be the duration of the calculation itself.
 - 2. The HOLD signal must obey its specified setup and hold times with respect to the CK2 clock (see the data sheet). Failure to do so can result in unpredictable operation.
 - 3. The Watchdog logic is not disabled in response to a HOLD request. It is the system designer's responsibility to ensure that the Watchdog function is not compromised.
 - 4. Interrupt conditions continue to be monitored during HOLD service. However, they are not serviced until the HOLD request is removed.

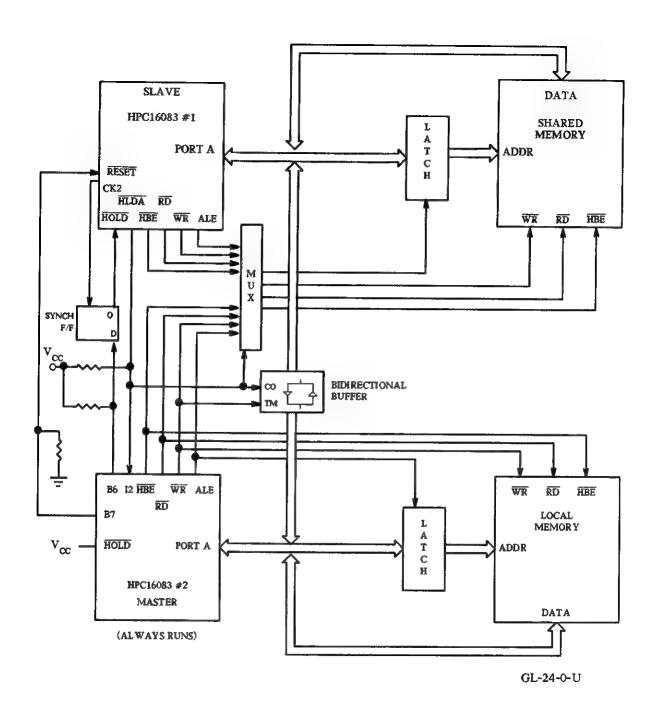


Figure 11-1. Shared Memory Application

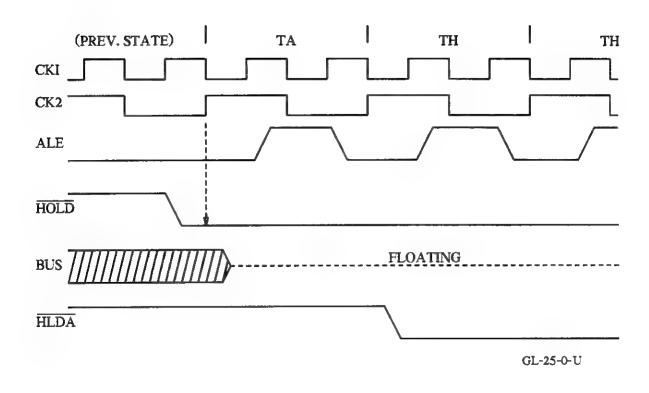


Figure 11-2. $\overline{\text{HOLD}}$ / $\overline{\text{HLDA}}$ Timing: Entry into Hold States

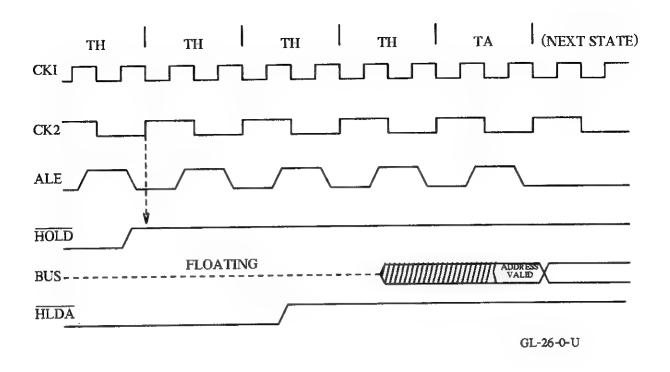


Figure 11-3. HOLD / HLDA Timing: Exit from Hold States

11.3 PIN CONFIGURATION

The RDY/HLD pin is assigned the HOLD function by clearing the RDY/HLD bit in the IRCD register. This is its initial condition on reset. The alternate function RDY (Section 6.5) can be selected instead by setting the RDY/HLD bit to 1.

The RDY/HLD pin has a weak internal pull-up device connected to it; leaving the pin open is sufficient to disable its functions.

The HLDA signal is the alternate function of Port B pin B7. It is enabled by setting bit 7 of both the BFUN and DIRB registers. Connecting an external pull-up resistor to this pin is recommended, since, like all Port B pins, it floats after reset until it is configured.

CAUTION

The RDY/HLD pin should never be allowed to remain low near the end of a RESET pulse; the HPC will fail to initialize the Stack Pointer, and will write to a random address (e.g., an I/O register) upon executing the RESET interrupt.

			\
		_	`
			,

RESET STATE

12.1 INTRODUCTION

The HPC16083 contains logic to initialize the state of the processor whenever a logic "0" is applied to the $\overline{\text{RESET}}$ pin. The logic "0" must be applied for at least sixteen cycles of the CK2 clock. If the $\overline{\text{WO}}$ pin (Watchdog Output) is being used there are further considerations (see Chapter 13).

The RESET pin is a high-impedance Schmitt trigger input pin. Placing a low level on the RESET pin re-initializes the on-chip logic. When it is brought high again, an interrupt is generated which causes the processor to jump to the address contained in the word at memory location FFFE.

The destination of the Reset jump must be located in the on-chip address space because the HPC powers up in Single Chip Mode (the EA bit in the PSW register is cleared on reset). This requirement may be disregarded in 16-Bit ROMless modes only, and then only if the program will always be run in that mode and the Watchdog logic will never be used.

The HBE pin (B12) is sampled on the rising edge of the RESET pulse, and determines whether the HPC is to enter 16-Bit Mode or 8-Bit Mode. The HBE pin has a weak internal pull-down device which is active only during the RESET pulse. This automatically configures the HPC in 16-Bit Mode if the HBE pin is left unconnected. If HBE is driven high externally, the HPC enters 8-Bit Mode.

The HPC16083 performs the following initializations at reset:

- The Stack Pointer is initialized to contain all zeros. A push is then performed so that the Stack Pointer contains 0002. Note that the push overwrites whatever was in addresses 0000 and 0001.
- In the PSW register, the bits EHI, HLT/IDL, EA, WAITO and WAIT1 are reset. This sets the number of pre-programmed Wait states to four, selects Single-Chip Mode, and selects Idle Mode as the power-down mode.
- The Direction Registers for Ports A and B are cleared, causing the associated port pins to float. External pull-up or pull-down resistors may be attached to these pins if it is necessary for them to go to a particular reset state.
- All bits in the BFUN register are cleared, selecting the Port I/O function for all Port B pins.
- In the UPIC register, the bits UPIEN and WRRDY are cleared. The RDRDY bit is set to 1.
- In the ENIR register, the Global Interrupt Enable (GIE) bit is reset, disabling all maskable interrupts. None of the other Interrupt Enable bits are affected.
- In the IRCD register, the RDY/HLD bit is cleared, selecting the HOLD function for the RDY/HLD pin. The uWMODE bit is cleared, selecting Slave Mode for the MICROWIRE interface (see Section 17.2). The other bits are not affected.

- The HALT Enable register at address 00DC is cleared. This prevents the Halt power-down mode from being entered until it is explicitly enabled by software (see Section 14.2).
- Timer T0 is reset to zero. The associated Watchdog logic is reset. If it has already triggered, however, the pulse on the WO pin will continue. The other timers are not affected.
- The PORTP register is cleared. This has the effect of selecting the Port P pins as bit-programmable outputs, and initializing them to zero levels. (Port P pins do not float.)
- UART registers ENU, ENUI and ENUR are cleared except for ENU bit TBMT, which is set.

WATCHDOG LOGIC

13.1 INTRODUCTION

The HPC16083 incorporates an independent block of logic, referred to as the Watchdog logic, to monitor operation of HPC programs. The Watchdog logic signals an error when a program attempts an out of bounds memory access or enters an infinite loop.

Illegal conditions detected by the Watchdog are brought to the system's attention by means of an associated pin named \overline{WO} . \overline{WO} is an open drain output, normally floating. Upon detection of an illegal condition it is pulled low by the Watchdog logic for 16—32 cycles of the CK2 clock and then returns to the floating state.

The system can use this output in any manner desired, or not at all. As an example, by connecting the \overline{WO} pin to the \overline{RESET} pin the HPC can be forced to reinitialize on triggering the Watchdog logic.

The $\overline{\text{WO}}$ pin is monitored as a feedback input through a Schmitt trigger circuit. The on-chip counter that times the pulse width is not started until a valid zero level is actually detected on the pin. This is done to guarantee the pulse width given above, even when driving a highly capacitive load (typical of the Reset circuit in many systems). In order for the $\overline{\text{WO}}$ pin to be triggered again, it must be pulled high externally (above the upper Schmitt threshold).

13.2 POTENTIALLY INFINITE LOOPS

The Watchdog logic detects potentially infinite loops based on Timer T0, which is clocked at a fixed CKI/16 rate. Since Timer T0 is being used as the time base for the Watchdog logic, the user is not allowed to write to this timer. Timer T0 is initialized to zero at reset.

In order to detect potentially infinite loops the Watchdog logic requires software to service it periodically. The Watchdog logic is serviced by writing the value 8421 (hexadecimal) as a word into the Watchdog register at address 0194. This must be done at least once every 65536 counts of Timer T0, and not more often than once every 4096 counts. Failure of software to do this causes a pulse to appear on \overline{WO} . The Watchdog register is a write-only register.

13.3 ILLEGAL ADDRESSES

The HPC16083 contains 8 Kbytes of ROM (addresses E000—FFFF) and 512 bytes of RAM and registers (addresses 0000—01FF). An address within either of these areas is considered logically on-chip. In Single-Chip Mode (Normal or ROMless, EA bit in PSW register is logic "0"), accesses outside these ranges are considered illegal, indicating that the program has encountered a fatal error. The Watchdog logic detects these conditions and pulses the $\overline{\text{WO}}$ pin low when they occur.

- NOTES: 1. The counter that times the duration of the \overline{WO} pulse is not affected by the \overline{RESET} signal. On a power-on reset, this counter comes up in a random state, and may cause a pulse on \overline{WO} . Therefore, when a Reset and a Watchdog event occur at the same time, or during a power-on reset, the \overline{WO} pulse must be allowed to "run its course" before the \overline{RESET} pulse is removed. In a system that uses \overline{WO} , then, the recommended minimum width of an externally-applied (i.e., non-Watchdog) \overline{RESET} pulse is 64 cycles of the CK2 clock.
 - 2. If the Watchdog feature is not being used, it is recommended (for power consumption reasons) that the \overline{WO} pin be tied permanently low.
 - 3. In Expanded Mode (EA bit "1"), there are no illegal addresses and the full 64 Kbytes (0000-FFFF) may be accessed without triggering a Watchdog Output.

POWER SAVE MODES OF OPERATION

14.1 INTRODUCTION

The HPC16083 can be caused to enter two power-save modes of operation:

- Halt Mode, in which all on-chip functions halt, including the crystal oscillator.
- Idle Mode, in which the on-chip oscillator, the Watchdog logic and Timer T0 continue to run. In this mode, the HPC is awakened automatically by overflows of the T0 timer.

14.2 HALT MODE

Halt Mode is a full shutdown of the HPC. All internal functions halt, and the HPC draws only a very small DC leakage current. The contents of the on-chip RAM are not affected. Unless Vcc is lowered during Halt Mode (see below), the I/O ports and on-chip registers are also not affected. The UART, however, is reset upon entering Halt Mode, and must be set up again when program execution continues.

Halt Mode has a security feature, because it invalidates Watchdog monitoring and should therefore not be entered accidentally. To enable Halt Mode, the following steps must be performed:

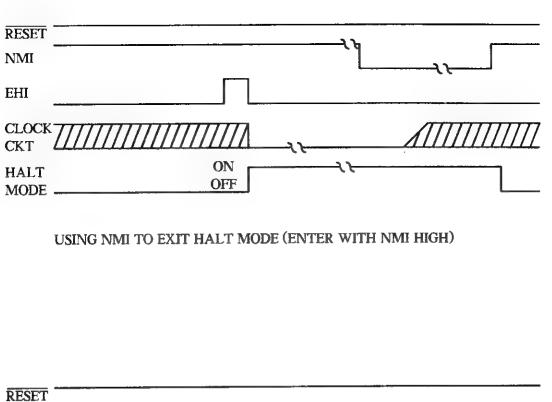
- Upon Reset, the program must write the value 8421 (hexadecimal) as a word into the Halt Enable register at address 00DC. Until this is done, Halt Mode is disabled. If a value other than 8421 is written, Halt Mode is permanently disabled until the next Reset. Any further writes into the Halt Enable register are ignored. Attempting to enter Halt Mode while it is disabled causes the HPC to enter Idle Mode instead. When the Halt Enable register is read, it presents a "1" in its least-significant bit if Halt Mode is enabled and a "0" if it is not.
- Halt Mode must be selected (as opposed to Idle Mode) by setting the PSW bit HLT/IDL to a "1". The default value of this bit on reset is "0".

Having done the above, the program is now allowed at any time to enter Halt Mode by setting the PSW bit EHI (Enable Halt/Idle) to a "1". It may also at any time alter the PSW HLT/IDL bit to switch between the Halt Mode and Idle Mode selections.

Entering Halt Mode clears the EHI bit in register PSW until it is set again by software. The HLT/\overline{IDL} bit is not affected by this.

There are two possible ways to exit Halt Mode. Activating the RESET pin will reset and restart the HPC. The Non-Maskable Interrupt may also be used for this purpose (see Figure 14-1).

Although the oscillator circuitry is brought to a stop in Halt Mode, the full state of the machine, including the on-chip RAM, registers and I/O ports, is preserved. When the HPC has entered Halt Mode, the voltage applied to the Vcc pins may be brought down below its operating voltage, to a minimum RAM keep-alive value (see the data sheet) to further decrease



RESET

NMI

EHI

CLOCK
CKT

HALT
MODE
OFF

USING NMI TO EXIT HALT MODE (ENTER WITH NMI LOW)

GL-27-0-U

Figure 14-1. HALT Timing: Using NMI to Exit HALT Mode

power consumption. When this is done, however, the contents of the on-chip registers are no longer guaranteed, and only the RAM contents are preserved. Vcc must be brought back to its full operating value before exiting Halt Mode, and only a Reset should be used to exit if Vcc has been reduced below the minimum operating voltage. Note that the Reset interrupt has the effect of overwriting the contents of RAM locations 0000 and 0001; these two bytes cannot be preserved.

A logic low level on the NMI pin (pin II) causes the HPC to restart its oscillator circuit. However, as long as NMI is held low the HPC does nothing else. NMI should be held low long enough to ensure that the oscillator stabilizes, as defined for the RESET signal on a power-on reset.

On the rising edge of NMI, the remainder of the HPC starts up and an NMI interrupt is generated. The HPC vectors to the NMI interrupt service routine and resumes normal operation from that point on.

The foregoing discussion assumes that the HPC16083 enters Halt Mode while maintaining a logic high at the NMI pin. Note however that one can elect to enter Halt Mode with the NMI pin held at ground. In this case, the HPC enters a power-save mode; the oscillator circuitry is not stopped, but all other logic on the chip is brought to a halt. This means that power consumption is higher than Halt Mode with NMI high, but the system can immediately exit Halt Mode without any oscillator start-up delay.

When NMI is used to exit from Halt Mode, the instruction immediately after the halting instruction is executed before the NMI interrupt is serviced. This happens because the instruction immediately after the halting instruction is already in the instruction queue.

Although all internal clocking has been stopped, the HPC continues to latch interrupt requests on the I2, I3 and I4 pins. These interrupt requests, however, do not cause the HPC to exit Halt Mode. If the NMI interrupt is used to bring the HPC out of Halt Mode, these interrupts are serviced after the NMI service routine has completed. If a Reset causes the HPC to exit Halt Mode, these interrupts are cleared.

14.2.1 State of Pins During Halt Mode

Single Chip Mode:

All I/O ports configured as outputs will retain the state they had previous to entering the Halt Mode. Port pins that are configured as inputs should be at V_{CC} or GROUND to achieve the lowest current drain.

Expanded or ROMless Modes:

Port A will have the address of the instruction which follows the instruction that sets the EHI bit in the PSW register. The bus controls, pins \overline{RD} and \overline{WR} , are high while ALE is low. The state of \overline{HBE} depends on whether the next address is from an even or odd location. The state of the remaining pins will be as described in the Single Chip Mode.

Timer Outputs (Port P, TSO-TS3, TSI0 and T3I0):

Pins configured as timer outputs will have the same state during the Halt Mode as before the Halt Mode was entered, they will change only after exiting the Halt

Mode. If a toggle signal from a timer output occurs at the same time as the Halt Mode is entered, the pin will not toggle until the Halt Mode is exited.

CK2: CK2 will be a high level during the Halt Mode.

CK0: CK0 will be high in the Halt Mode.

W0: The Watchdog logic stops during Halt Mode and no Watchdog Output (W0) can occur. If W0 is not used, the pin should be tied low to minimize power consumption.

14.3 IDLE MODE

Idle Mode is a partial shutdown of the HPC. The crystal oscillator, Timer T0 and the Watchdog logic continue to operate, but all other on-chip logic is stopped. The states of the I/O ports and the contents of the on-board RAM are not altered in any manner. The UART is reset upon entering Idle Mode, and must be re-initialized when program execution continues.

To place the HPC16083 in Idle Mode, a program does the following:

- ullet The program selects Idle Mode (as opposed to Halt Mode) by resetting the HLT/ $\overline{\text{IDL}}$ bit in the PSW register.
- The program triggers Idle Mode by setting the EHI bit in the PSW register.

Entering Idle Mode clears the EHI bit in register PSW until it is set again by software. The HLT/IDL bit is not affected.

Unlike Halt Mode, the Vcc voltage cannot be decreased below the minimum operating voltage while in Idle Mode. Power savings in Idle Mode are a result of stopping the internal chip circuitry.

There are three means by which the HPC can exit Idle Mode:

- Pulling the RESET pin to ground causes the HPC to re-initialize and restart.
- A rising edge on the NMI pin causes the HPC to exit Idle Mode and generates an interrupt.
- Any overflow of Timer T0 causes the HPC to exit Idle Mode.

The instruction immediately following the one that triggers Idle Mode is always executed before any pending interrupts are serviced, since it is already in the instruction queue. If the Timer TO overflow brings the HPC out of Idle Mode, the HPC resumes operation from the instruction following the one that placed the HPC in Idle Mode.

Although most internal clocking has been stopped, the HPC continues to latch interrupt requests from the I2, I3 and I4 pins. These interrupt requests, however, do not cause the HPC to exit Idle Mode. They are serviced only after the HPC has been brought out of Idle Mode by one of the events listed above. If a Reset is used to bring the HPC out of Idle Mode, these interrupt requests are not serviced, but are cleared instead.

The Watchdog logic is active during the time the HPC16083 is in Idle Mode, and it must still be serviced to avoid a Watchdog event. Servicing it once every time Timer TO overflows is sufficient to accomplish this.

NOTE: The Timer TO interrupt should not be enabled when Idle Mode is entered.

This interrupt is not always posted on exit from Idle Mode.

14.3.1 State of Pins During Idle Mode

Single Chip Mode:

All I/O ports that are configured as outputs will retain the same state as previous to entering the Idle Mode. Port pins that are configured as inputs should be at V_{CC} or GROUND to achieve the lowest current drain.

Expanded or ROMless Modes:

Port A will have the address of the instruction which follows the instruction that sets the EHI bit in the PSW register. The bus control pins \overline{RD} and \overline{WR} are high while ALE is low. The state of \overline{HBE} depends on whether the next address is from an even or odd location. The state of the remaining pins will be as described in the Single Chip Mode.

Timer Outputs (Port P, TSO-TS3, TSI0 and T310):

Pins configured as timer outputs will have the same state during the Idle Mode as before the Idle Mode was entered, they will change only after exiting the Idle Mode. If a toggle signal from a timer output occurs at the same time as the Idle Mode is entered, the pin will not toggle until the Idle Mode is exited.

CK2: CK2 will be at a high level during the Idle Mode.

CKO: CKO will be oscillating at the crytstal or input clock frequency during Idle Mode.

W0: The Watchdog logic continues to function during the Idle Mode and the Watchdog Output (WO) pin will operate just like it does during normal operation (see Chapter 13).

		<u> </u>
		`

Chapter 15

INTERRUPTS

15.1 INTRODUCTION

The HPC16083 supports vectored interrupts from eight sources. These sources are the \overline{RESET} pin, the NMI pin (I1), the Port I pins I2, I3 and I4, the EI pin, the Timer section and the UART (including the pin \overline{EXU} ; see below).

The HPC contains arbitration logic to decide which interrupt will be serviced first if two or more interrupts occur simultaneously. The interrupt source with the highest arbitration ranking is serviced first; Table 15-1 lists the interrupt sources by their arbitration ranking.

TABLE 15-1. INTERRUPT ARBITRATION

VECTOR ADDRESS	INTERRUPT SOURCE	ARBITRATION RANKING		
FFFE	Reset: resets logic while low, triggers interrupt on rising edge.	0 (Highest)		
FFFC	Non-Maskable: external, on rising edge of I1 pin	1		
FFFA	External, on rising/falling edge of I2 pin	2		
FFF8	External, on rising/falling edge of I3 pin	3		
FFF6	External, on rising/falling edge of 14 pin	4		
FFF4	Internal, from Timers	5		
FFF2	Internal, from the UART, or external, on EXUI pin low	6		
FFF0	External, on rising/falling edge or high/low level on EI pin.	7 (Lowest)		

RESET and NMl are non-maskable interrupts. The maskable interrupts can be individually enabled or disabled. The control register ENIR (Section 15.5) contains a bit corresponding to each of the maskable interrupt sources. Setting the bit for an interrupt source enables interrupts from it while resetting the bit disables all interrupts from it.

The HPC16083 also permits the maskable interrupts to be globally enabled or disabled by means of the Global Interrupt Enable (GIE) bit in the ENIR register. Resetting the GIE bit disables all maskable interrupts. Setting it enables all maskable interrupts whose corresponding ENIR bits are also set. The GIE bit is initialized to zero on reset.

RESET is a low-active, level-sensitive interrupt. The interrupts from NMI, I2, I3 and I4 are edge sensitive. NMI is sensitive to rising edges on the pin. The external maskable interrupts I2, I3 and I4 may be programmed to be sensitive to either rising or falling edges through corresponding bits in the IRCD control register (Section 15.5). The EI pin is an interrupt that may be programmed using the EICON register to be high or low level sensitive or rising or falling edge sensitive. The on-chip peripheral interrupts (Timers and UART) are level sensitive; once asserted, they remain so until they are serviced by appropriate action at the peripheral, or until they are disabled at the peripheral. The pin EXUI is a level-sensitive, low-active interrupt, which may be used as an additional source of the UART interrupt. If it is not being used, it should be left disconnected, allowing an on-chip pull-up device to hold it inactive.

The HPC16083 recognizes eight interrupt vectors, located at the end of memory. Each vector is a 16-bit word, filled by the programmer (using appropriate directives to the assembler or compiler) with the address of the service routine for its interrupt. Interrupts from the same on-chip peripheral share the same interrupt vector; for example, both the UART transmitter and the receiver can interrupt the processor independently, but both interrupts vector to the same address. The mapping between the interrupt sources and the interrupt vectors is shown in Table 15-1. All interrupt service routines, except the Reset and NMI routines, may be located anywhere in the full 64 Kbytes of address space. The Reset service routine must be located in the on-chip address space, because the HPC initializes itself to Single-Chip Mode on reset. The NMI service routine should also be placed in the on-chip address space unless the system guarantees that NMI cannot happen shortly after a reset.

15.2 INTERRUPT PROCESSING

With the exception of the Reset interrupt, which is serviced immediately, all interrupts are serviced after the HPC completes the current instruction.

Upon receipt of an interrupt, the HPC copies the Global Interrupt Enable (GIE) bit of the ENIR register into the CGIE bit of the PSW register. The GIE bit is then reset to disable all further maskable interrupts. The return address is pushed onto the stack, incrementing the Stack Pointer by two. The Program Counter is then loaded with the contents of the vector location corresponding to the particular interrupt.

Interrupts may be nested by setting the GIE bit within the interrupt service routine; otherwise the GIE bit is automatically set, and interrupts re-enabled, only upon executing the RETI (Return from Interrupt) instruction. This instruction also pops the return address from the stack into the Program Counter, and causes the processor to resume execution of the interrupted routine.

The CGIE bit in the PSW register is intended for use in returning from the Nonmaskable Interrupt (NMI). To return, the NMI service routine should check the CGIE bit in the PSW register, and should use RETI if that bit is set, or simply RET if it is clear.

15.3 INTERRUPT CONTROL REGISTERS

The HPC16083 contains four registers to control interrupt handling. These registers are called ENIR (Interrupt Enable), IRCD (Interrupt Condition), EICON (EI Configuration), and IRPD (Interrupts Pending).

The ENIR register contains individual enable bits for all maskable interrupts, and the GIE (Global Interrupt Enable) bit, which serves to disable all maskable interrupts while interrupt service is in progress.

The IRCD register programs the type of edge (rising or falling) to be used to trigger the I2, I3 and I4 interrupts. This edge is also used to trigger the corresponding capture registers in the Timer section (see Section 16.2). The IRCD register also contains configuration bits for other functions unrelated to interrupt handling.

The EICON register programs the four types of interrupt triggers possible for the EI pin (rising or falling edge and high or low level). It also contains an Acknowledge bit for clearing the EI pending bit in the IRPD register after an edge sensitive interrupt is latched.

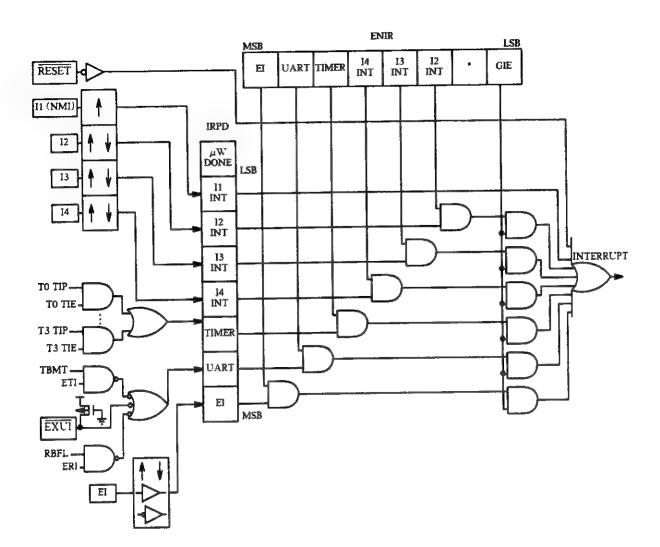
The IRPD register indicates which interrupts are pending for service. This register contains a bit for each interrupt vector. The occurrence of a specific interrupt trigger condition causes the corresponding IRPD bit to be set. These bits are set regardless of whether the corresponding interrupts are enabled. The pending flags corresponding to the edge-triggered interrupts I2, I3 and I4 are automatically cleared by the HPC on entering their service routines. The EI bit in the IRPD register is set to one after an edge sensitive EI interrupt is latched. The bit is not automatically reset on entering the service routine and must be reset by writing a zero to the ACK bit (bit 2) in the EICON register. When EI is programmed to detect high or low levels, the EI bit is high as long as the active level remains on the EI pin.

The IRPD register's NMI, UART, Timer and EI bits are read only. The uWDONE bit is also read-only, but does not correspond to an interrupt. The bits in register IRPD from on-chip peripherals (Timers and UART) directly reflect the contents of the interrupt pending bits from those peripherals. Servicing the interrupt condition in the peripheral section automatically clears the corresponding bit in the IRPD register. The INT2, INT3, and INT4 bits are designed so as only to allow a zero to be written into them by software. Writing a one has no effect on them. Clearing one of these three bits has the effect of removing the corresponding interrupt request also.

NOTE: Never use a bit instruction (SET or RESET), or any other instruction that performs a read/modify/write access, to clear a bit in the IRPD register. These can have the side effect of clearing another IRPD bit that was being set by hardware at the same time. It is recommended instead that a simple Load instruction (LD) be used, to write a mask value consisting of ones in all bit positions except those for which the corresponding IRPD bits are to be cleared.

In applications not using interrupts, software can read the IRPD register to allow a software-based priority scheme to be implemented.

Figure 15-1 presents the roles of these registers in block diagram form.



GL-28-0-L

Figure 15-1. Block Diagram of Interrupt Logic

15.4 INTERRUPT LATENCY

Interrupts are serviced by HPC hardware only after the current instruction finishes execution. The time that the HPC needs to fetch the interrupt vector and to push the return address is (11 + 3 W_i + W_d) cycles of the CK2 clock. The maximum latency time is the sum of the longest instruction plus (11 + 3 W_i + W_d) CK2 cycles, totalling approximately 10 microseconds at 17 MHz. This assumes that a Multiply or Divide instruction is executing at the time the interrupt is presented. Typical HPC instructions are much shorter. See Appendix C for instruction execution times, and Sections 5.2 and C.4 for information on W_i .

15.5 INTERRUPT CONTROL REGISTER MAPS

The interrupt control registers are diagrammed below.

15.5.1 ENIR — Interrupt Enable Register

EI. X	UART X	TIMERS X	INT4 X	INT3 X	INT2 X	*	GIE 0
	·	V • W	Byte a	t 00D0			-

X State is indeterminate on reset. 0 Cleared on reset. ΕI Enables interrupts from the External Interrupt (EI) pin while set. UART Enables interrupts from the UART while set. **TIMERS** Enables interrupts from the timers while set. INT4 Enables interrupts on the pin I4 while set. INT3 Enables interrupts on the pin I3 while set. INT2 Enables interrupts on the pin I2 while set. Unassigned bit. Should be considered indeterminate data. GIE Global enable for all maskable interrupts. Maskable interrupts are enabled only if this bit is also set.

15.5.2 IRCD — Interrupt Condition Register

**	##	**	I4 POL	I3 POL	I2 POL	uWMODE	RDY/HLD
0	0	0	0	0	0	0	0
			Byte a	t 00D4			

Cleared on reset. 0 Unassigned bit. Should be considered indeterminate data. 丰本 Selects the polarity of the edge on the I4 pin which triggers an interrupt 14 POL and triggers Capture Register I4CR in the Timer section. A rising edge is selected if the bit is set and a falling edge is selected if the bit is reset. Selects the polarity of the edge on I3 pin to trigger an interrupt and a I3 POL capture, as in I4 above. Selects the polarity of the edge on I2 pin to trigger an interrupt and a I2 POL capture, as in I4 above. Selects the mode of operation for the MICROWIRE/PLUS interface. uW MODE Setting this bit selects Master mode, and resetting it selects Slave mode. See Section 17.2. Selects the function of the RDY/\overline{HLD} pin. If this bit is set the pin has the RDY/HLD READY function interpretation (Section 10.5), and if it is reset the HOLD function is selected (Chapter 11).

15.5.3 EICON — EI Configuration Register

*	*	*	*	*	ACK 1W	MODE 0	POL 0
			Byte a	t 015C			1

This bit is set on Reset.
This bit is cleared to zero on chip Reset.
This is a Write-Only bit and reads back as its initial state.

An undefined bit that cannot be written to and is read back as a one.

EI interrupt acknowledge bit. Writing a zero to this bit clears the EI ACK

interrupt pending bit in the IRPD register after an edge sensitive

interrupt has been latched.

MODE This bit defines whether the EI interrupt function is level or edge

sensitive (see Table 15-2).

POL This bit selects the active polarity or edge of the EI interrupt and is

dependent on the MODE bit, see Table 15-2.

TABLE 15-2. EICON REGISTER (BITS 1 AND 0)

MODE BIT	POL BIT	INTERRUPT FUNCTION	INPUT CAPTURE FUNCTION		
0	0	high level	falling edge		
0	1	low level	rising edge		
1	0	falling edge	falling edge		
1	1	rising edge	rising edge		

WARNING

False El interrupts can occur if the following conditions are met.

- 1. The MODE and POL bits are changed to detect falling edge interrupt on EI pin while a low level signal exists on EI pin.
- 2. The MODE and POL bits are changed to detect rising edge interrupt on El pin while a high level signal exists on EI pin.

The user can avoid servicing a false EI interrupt by: disabling the EI interrupt (writing zero to the EI bit in ENIR register), changing MODE and POL bits, clearing false EI interrupt pending bit in IRPD register (writing zero to ACK bit in EICON register) and then re-enabling the El interrupt (writing a one to EI bit in ENIR register).

In edge triggered modes, the interrupt is set pending when the associated capture occurs in the EICR register, this can be up to 8 CK2 periods later than the edge.

15.5.4 IRPD — Interrupt Pending Register

EI	UART	TIMERS	INT4	INT3	INT2	NMI	uWDONE
XR	XR	XR	XC	XC	XC	XR	XR
			Byte a	t 00D2			

X State is indeterminate on reset. R Bit is read-only. C Bit can be cleared by writing zero to it; writing a one does nothing. EI When programmed to detect edges, this read-only bit is set to one after a rising or falling edge is detected. The EI bit is reset by writing a zero to the ACK bit in the EICON register (see Section 15.5.3); this bit is not automatically cleared by servicing the interrupt. When EI is programmed to detect high or low levels, the EI bit is high (logic "1") as long as the active level remains on the EI pin. This read-only bit goes high when an interrupt is requested by the UART UART. This generally occurs when a character is either transmitted or received. It is cleared when the interrupt condition is serviced; e.g., the receive buffer RBUF is read or the transmit buffer TBUF is written into. See Section 18.6. This read-only bit is a level indicating, when high, that an underflow TIMERS interrupt is pending from a timer T1-T7, or an overflow interrupt is pending from timer T0. It is cleared by acknowledging the interrupt in the TMMODE or PWMODE register (Section 16.4). This bit is set when the edge selected for I4 occurs. It is to be reset by INT4 writing a zero into it, or by allowing its interrupt service routine to be entered (automatically clearing the bit). Writing a one to this bit has no effect. This bit is set when the edge selected for I3 occurs. It is to be reset by INT3 writing a zero into it, or by allowing its interrupt service routine to be entered (automatically clearing the bit). Writing a one to this bit has no effect. This bit is set when the edge selected for I2 occurs. It is to be reset by INT2 writing a zero into it, or by allowing its interrupt service routine to be entered (automatically clearing the bit). Writing a one to this bit has no effect. This bit is set when a rising edge occurs on the NMI (11) pin. This is a **NMI** read-only bit, and is automatically reset when the NMI service routine is entered. This bit is set when the MICROWIRE/PLUS circuitry completes an uW DONE eight-bit transfer. It is reset by accessing the SIO register. This is a polled flag bit only, and DOES NOT GENERATE AN INTERRUPT.

Chapter 16

TIMERS

16.1 INTRODUCTION

The HPC16083 contains nine 16-bit timers, T0 through T8. Timer T0 is an up-counter, with up to three associated capture registers I2CR, I3CR and I4CR. (I3CR and I2CR serve as Timer T1 and its input register R1, respectively, when configured to do so.) Timer T8 contains the same value as T0 at all times. T8 serves as the time base for the fourth input capture register EICR. Timers T1 through T7 are down-counters with associated input registers, R1 through R7, from which they are loaded automatically when they underflow. Timers T2 through T7 have individual output signals (T2 controlling up to five outputs). Timers T2 and T3 may be clocked externally through two separate inputs.

The HPC16083 permits all timers other than Timer T0 and T8 to be started or stopped at any time. These timers and their input registers (again excluding T0 and T8) may also be read at any time. Timer contents are guaranteed valid whenever they are read by software, even if the timer is being clocked from an asynchronous external signal. Software writes to an input register or timer are legal at any time, whether its timer is running or not. Loading a timer (automatically or by software) does not affect the contents of the input register; starting a timer and leaving the input register alone causes a timer underflow to occur at regular intervals determined by the contents of the input register. The timer is loaded automatically at the point that it would have underflowed from 0000 to FFFF. Note, that in order to get division modulo n from a timer, its input register must be loaded with the value n-1.

All timers are capable of generating interrupts through the TIMERS vector (Chapter 15).

16.2 TIMER OPERATIONS

16.2.1 Timers T0 and T8

Timer T0 is a free-running timer, clocked at a fixed CKI/16 rate. Its primary use by software is in making time interval measurements. However, Timer T0 also forms the time reference for the Watchdog logic and provides a means of exiting the Idle power-save mode. For these reasons, it is not possible to stop Timer T0 (except by entering the Halt power-save mode) or to write to it. It may, however, be read indirectly by commanding the Capture Register I4CR (see below) to load continuously from it. Timer T0 has no associated output pins, but it can be programmed to generate an interrupt on overflow. Bits 0—3 of the TMMODE register monitor Timer T0 overflow events and control interrupt generation. The TMMODE register is described in Section 16.4.

Timer TO has associated with it three 16-bit Capture Registers, called I2CR, I3CR and I4CR. These registers load from Timer TO upon seeing a programmed edge on their corresponding pins: I2, I3 or I4. The selection of the edge polarity for each pin is made in the IRCD register, as described in Section 16.4. These three pins are also capable of generating interrupts on the same edges (see Figure 16-1). In order for them to trigger a capture, the appropriate interrupt enable

bit (12, 13, 14) must be set in the ENIR register. (It is not, however, necessary for the GIE bit to be set.) Two of the three capture registers, I3CR and I2CR, may be reconfigured to form, respectively, Timer T1 and its associated input register R1. The three capture registers are read/write registers.

Capture register I4CR can also be programmed to load continuously from Timer T0, regardless of the I4 input pin. This feature, which provides the only means of inspecting Timer T0, is enabled by setting the C4F (Capture 4 Function) bit in the T0CON (T0 Configuration) register. I4CR is loaded once every eight cycles of the CK2 clock; hence a program wishing to set C4F and then examine Timer T0 must wait this period of time after setting C4F to ensure that the I4CR register has been loaded at least once. Placing two NOP instructions after the instruction that sets the C4F bit provides a sufficient delay.

Timer T8 counts up like T0 and serves as the time base for the fourth input capture register EICR (see Figure 16-1). The EICR capture register is loaded with the value in timer T8 upon seeing a programmed edge on the EI pin. Timer T8 always contains the same value as timer T0. For this reason, you can think of all four input capture registers (I2CR, I3CR, I4CR, and EICR) being loaded with the value in timer T0 when edges are detected on pins I2, I3, I4, and EI. The EI pin may be programmed to detect rising or falling edges and high or low levels for generating interrupts. These four possible configurations for the EI pin are selected by programming two bits in the EICON register (see Section 16.4). Only rising or falling edges are detected when using the EI pin with the EICR register for measuring time intervals (input capture). When using the EI pin for input capture, the EI pending bit in the IRPD register must be cleared by writing a zero to the ACK bit in the EICON register and the EI interrupt must be enabled by setting the EI bit in the ENIR register.

16.2.2 Timer T1

Timer T1 and its associated input register R1 are available when they are not configured as the T0 Capture Registers I3CR and I2CR (see Figure 16-1). The configuration is selected by the state of the T0 ORG (T0 Organization) bit in the T0CON register: T0 ORG = 0 selects the Capture Register functions I2CR and I3CR, and T0 ORG = 1 selects the Timer functions. Timer T1 counts down at a constant rate of CKI/16 while enabled. It has no associated output pin, but can be enabled to trigger an interrupt on underflow. Timer T1 is controlled through bits 4—7 of the TMMODE register, described in Section 16.4.

16.2.3 Timers T2 and T3

Timers T2 and T3 are the most flexible of the HPC timers (see Figure 16-2). The clock inputs to T2 and T3 may be independently selected as coming from one of 14 prescaled versions of the CKI clock, or from an external pin, as specified in the DIVBY register (see Section 16.4). Timer T2 can also be specified to be clocked on underflows from Timer T3, by appropriate selection in the DIVBY register; the pair then form in effect a single counter of up to 32 bits (depending on the value held in register R3). The external inputs to Timers T2 and T3 (if selected in the DIVBY register) are taken from Port B pins B3 and B4, respectively, and the timers are

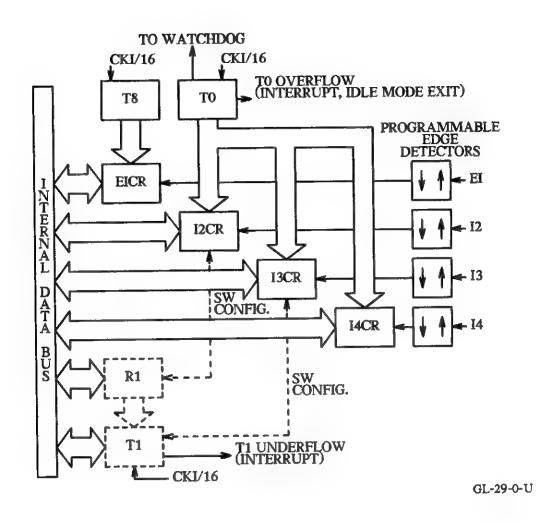


Figure 16-1. Timer TO, T1 and T8 With Four Input Capture Registers

decremented on falling edges. Since these pins can also serve as port I/O pins or timer outputs, they must also be configured correctly in the BFUN and DIRB registers for their alternate Timer functions, as given in Table 16-1 and Section 16.5. Timer outputs are discussed in Section 16.3.

Timer T2 is controlled through bits 8—11 of the TMMODE register, and Timer T3 is controlled through bits 12—15. See Section 16.4 for a description of the TMMODE control bits and their functions.

16.2.4 PWM Timers: T4-T7

Timers T4 through T7 are referred to collectively as the Pulse Width Modulation (PWM) Timers (see Figure 16-3). They count down at a constant rate of CKI/16 while enabled to do so, and they load from their associated input registers R4—R7 on underflow. Each timer has an associated output pin, as described in Section 16.3. and can be enabled to trigger an interrupt on underflow. Timers T4 through T7 are controlled through the PWMODE register, which is described in Section 16.4.

16.3 TIMER OUTPUTS

Timers T2 through T7 support generation of output signals.

Timer T2 has five output pins, which can be independently selected as outputs: B3 (called T2IO), B8 (TS0), B9 (TS1), B13 (TS2) and B14 (TS3). The pins TS0—TS3 are also referred to collectively as the "Timer Synchronous" outputs. Timer T3 has the single output pin B4 (T3IO). Timers T4 through T7 have one output each, on pins P0 (T4OUT) through P3 (T7OUT).

While one of these pins is configured for its alternate Timer function, its port output bit (in the PORTB register for T2 and T3, and in the PORTP register for T4—T7) toggles on every underflow of the corresponding timer. If the pin has been correctly configured as an output (Port B pins only; Port P pins are always outputs), the pin follows the value of the port output bit. The bit also becomes read-only to software, to avoid unintentional interference by a program modifying other unrelated bits in the same register.

See also Section 9.3 for details of the Port B pins and Section 9.6 for the Port P pins. Details of configuring these pins are given in Section 16.5.

NOTE: Outputs from different timers do not change at the same exact time. This is because the timers do not count simultaneously; they are organized into two independent groups of four (TO—T3 and T4—T7), and, within each group, one timer updates in sequence every two counts of the CK2 clock. The five outputs TSO—TS3 and B3 (T2IO), however, are exactly synchronous, since they are all derived from Timer T2.

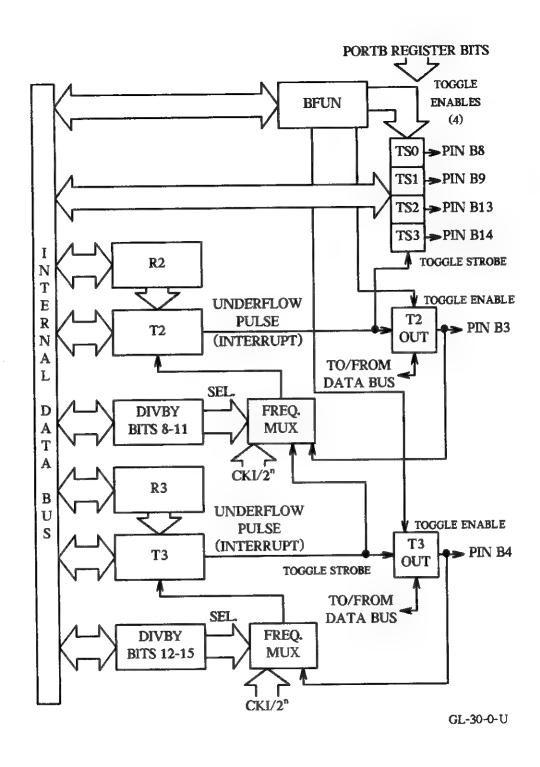
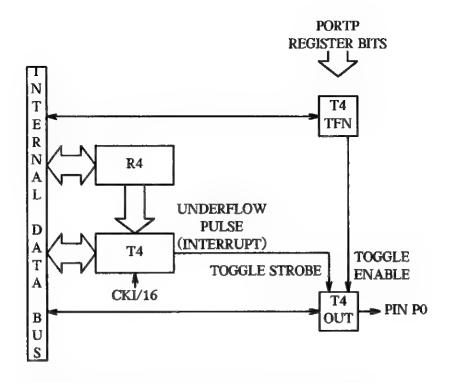


Figure 16-2. Timer Block Diagram: Timers T2 and T3



NOTE: Only Timer T4 is shown; T5, T6, and T7 are identical.

GL-31-0-U

Figure 16-3. PWM Timers (T4-T7) Block Diagram

16.4 TIMER SECTION REGISTERS

A set of fourteen registers are used to program the HPC16083 timers. The function and setup of these registers are described here. With the exception of the PORTP register, which is cleared, none of these registers are affected by a Reset, and their initial state on a power-on reset is undefined.

The PORTB, DIRB and BFUN registers control the outputs from and external inputs to Timers T2 and T3. However, because they have many other functions unrelated to the timers, they are described elsewhere, in Section 9.3.

I4CR

A 16-bit Capture Register at address 0180, triggered to load from Timer TO on a programmed edge on the I4 pin. It may also be programmed to load continuously from Timer TO, allowing Timer TO to be monitored through it. This option is controlled through the C4F bit in register TOCON.

I3CR/T1

A 16-bit Capture Register at address 0182, triggered to load from Timer T0 on a programmed edge on the I3 pin. While the bit T0 ORG in register T0CON is set, this register serves as Timer T1.

A 16-bit Capture Register at address 0184, triggered to load from Timer I2CR/R1 TO on a programmed edge on the I2 pin. While the bit TO ORG in register TOCON is set, this register serves as the input register R1 for Timer T1. A 16-bit Capture Register at address 015E, triggered to load from Timer **EICR** T8 on a programmed edge on the EI pin. This register is a read-only register. R2 The input register for Timer T2, at address 0186. Timer T2, at address 0188. **T2** The input register for Timer T3, at address 018A. R3 Timer T3, at address 018C. **T3** Divide-By Register; selects the clock inputs to Timers T2 and T3, to the DIVBY UART and to the MICROWIRE interface. **UART** uWIRE **T3 T2** Byte at 018E Byte at 018F Word at 018E

The formats of the DIVBY register fields are given below:

The clock to Timer T3 is programmed through DIVBY register bits 12—15 as indicated below.

BIT 15	BIT 14	ВГГ 13	BIT 12	CLOCK INPUT TO T3
0	0	0	0	External (pin B4)
0	0	0	1	T3 Output (stops T3)
0	0	1	0	CKI/16
0	0	1	1	CKI/32
0	1	0	0	CKI/64
0	1	0	1	CKI/128
0	1	1	0	CKI/256
0	1	1	1	CKI/512
1	0	0	0	CKI/1024
1	0	0	1	CKI/2048
1	0	1	0	CKI/4096
1	0	1	1	CKI/8192
1	1	0	0	CKI/16384
1	1	0	1	CKI/32768
1	1	1	0	CKI/65536
1	1	1	1	CKI/131072

The clock to Timer T2 is programmed through DIVBY register bits 8—11 as indicated below.

BIT 11	BIT 10	BIT 9	BIT 8	CLOCK INPUT TO T2
0	0	0	0	External (pin B3)
0	0	0	1	T3 Output
0	0	1	0	CKI/16
0	0	1	1	CKI/32
0	1	0	0	CKI/64
0	1	0	1	CKI/128
0	1	1	0	CKI/256
0	1	1	, 1	CKI/512
1	0	0	0	CKI/1024
1	0	0	1	CKI/2048
1	0	1	0	CKI/4096
1	0	1	1	CKI/8192
1	1	0	0	CKI/16384
1	1	0	1	CKI/32768
1	1	1	0	CKI/65536
1	1	1	1	CKI/131072

UART Bits 4—7 of the DIVBY register are used to select the UART baud rate clock. The possible values are listed below (see Section 18.7).

BIT 7	BIT 6	BIT 5	BIT 4	BAUD CLOCK (x16 Clock)	BAUD RATE 9.83 Mhz Crystal	BAUD RATE 16.88 Mhz Crystal
0	0	0	0	1	- Not Allowed -	The state of the s
0	0	0	1	<-Defined	by Timer T3 und	lerflow->
0	0	1	0	CKI / 16	38400	65536
0	0	1	1	CKI / 32	19200	32768
0	1	0	0	CKI / 64	9600	16384
o	1	0	1	CKI / 128	4800	8192
0	1	1	0	CKI / 256	2400	4096
0	1	1	1	CKI / 512	1200	2048
1	0	0	0	CKI / 1024	600	1024
1	0	0	1	CKI / 2048	300	512
1	0	1	0	CKI / 4096	150	256
1	0	1	1	CKI / 8192	75	128
1	1	0	0	CKI / 16384	38	64
1	1	0	1	CKI / 32768	19	32
1	1	1	0	CKI / 65536	9.4	16
1	1	1	1	CKI / 131072	4.7	8

uWIRE Bits 0—3 program the MICROWIRE serial clock (SK) frequency of operation when the HPC is in Master Mode (see Section 17.2).

BIT 3	BIT 2	BIT 1	BIT 0	MICROWIRE CLOCK			
0	0	0	0	Not Allowed			
0	0	0	1	Defined by Timer T3 underflows			
1		· ·		(Restricted; see Section 17.2.)			
0	0	1	0	CKI / 16			
0	0	1	1	CKI / 32			
0	1	0	0	CKI / 64			
0	1	0	1	CKI / 128			
0	1	1	0	CKI / 256			
0	1	1	1	CKI / 512			
1	0	0	0	CKI / 1024			
1	0	0	1	CKI / 2048			
1	0	1	0	CKI / 4096			
1	0	1	1	CKI / 8192			
1	1	0	0	CKI / 16384			
1	1	0	1	CKI / 32768			
1	1	1	0	CKI / 65536			
1	1	1	1	CKI / 131072			

TOCON Timer TO Configuration Register.

*	*	*	*	TO ORG X	TO C4F X	*	*
			Pute a	t 0192			

- X State of this bit indeterminate on Reset.
- * Unassigned bit. Should be considered indeterminate data.
- TO C4F Used to specify the function of the capture register I4CR.
 - TO C4F = 1 Conditions I4CR to load continuously from Timer T0.
 - TO C4F = 0 Conditions I4CR to capture events on Pin I4.
- TO ORG Used to specify the organization of the Timer TO capture registers I2CR and I3CR.
 - TO ORG = 1 Redefines I2CR to be Register R1 and I3CR to be Timer T1.
 - TO ORG = 0 Defines I2CR and I3CR to capture events on Pins I2 and I3.

EICON El Configuration Register; controls the interrupt conditions of the El pin.

*	*	*	*	*	ACK 1W	MODE 0	POL 0
			Ryte a	t 015C			-

1 This bit is set on Reset.

O This bit is cleared to zero on chip Reset.

W This is a Write-Only bit and reads back as its initial state.

Unassigned bit. Should be considered indeterminate data.

ACK EI interrupt acknowledge bit. Writing a zero to this bit clears the EI interrupt pending bit in the IRPD register (see Section 15.5.4) after an

edge sensitive interrupt has been latched.

MODE This bit defines whether the El interrupt function is level or edge

sensitive (see the following table).

POL This bit selects the active polarity or edge of the EI interrupt and is

dependent on the MODE bit, see Table 15-2.

	EICON	REGISTER (BIT	'S 1 AND 0)
MODE BIT	POL BIT	INTERRUPT FUNCTION	INPUT CAPTURE FUNCTION
0	0	high level	falling edge
0	1	low level	rising edge
1	0	falling edge	falling edge
1	1	rising edge	rising edge

TMMODE Timer Mode Register; controls Timers T0 — T3.

1						7	Word a	at 019	0						
T3 ACK OW	T3 STP X	T3 TIP XR	T3 TIE X	T2 ACK OW	T2 STP X	T2 TIP XR	T2 TIE X	T1 ACK OW	T1 STP X	T1 TIP XR	T1 TIE X	TO ACK OW	*	TO TIP XR	TO TIE X
	Byte at 0191									,	Byte a	t 0190)		

PWMODE Pulse Width Timer Mode Register; controls Timers T4 — T7.

ı	Word at 0150														
T7 ACK 0W	T4 STP X	T7 TIP XR	T7 TIE X	T6 ACK OW	T6 STP X	T6 TIP XR	T6 TIE X	T5 ACK OW	T5 STP X	T5 TIP XR	T5 TIE X	T4 ACK OW	T4 STP X	T4 TIP XR	T4 TIE X
	Ryte at 0151]	Byte a	t 0150)			

O Cleared on Reset.

X State indeterminate on Reset.

W Write-only (reads back as its initial state always).

R Read-only.

* An unassigned bit. Should be considered indeterminate data.

TIE Enables or disables the interrupts from the particular timer. If this bit is set the interrupt from the corresponding timer is enabled.

TIP Flags an interrupt pending from the particular timer. The interrupt request remains present as long as this bit and the TIE bit are both

Flags an interrupt pending from the particular timer. The interrupt request remains present as long as this bit and the TIE bit are both set. TIP is a read-only bit; it is set by underflow/overflow of the timer and is reset by writing a one to the associated ACK control bit (below). The TIP bit is set on timer underflow/overflow regardless of the state of the TIE bit.

STP Used to start or stop the associated timer.

STP = 1 The timer stops.

STP = 0 The timer operates continuously.

ACK

The Timer Interrupt Acknowledge control bit. Writing a one to this bit clears the associated Timer Interrupt Pending bit (TIP). The ACK bit is write-only; zero is always read back from it.

ACK = 1 Resets the TIP bit.

ACK = 0 No action.

There is a pipeline delay of eight CK2 clock periods between the point that NOTE: a timer underflow or overflow is posted internally and the point that the TIP flag bit is set. An important implication of this is that setting the ACK bit and the STP bit in the same access is not always sufficient to guarantee that the TIP bit will stay cleared: if the STP and ACK bits are set immediately after the timer has underflowed, the TIP bit will be set by hardware after the point that it was cleared by the ACK bit. Note especially that this situation is always relevant when initializing the TMMODE and PWMODE registers after a power-on reset, since the states of the timers and their control registers are indeterminate at that time. To ensure that the TIP flag bit remains cleared, it is recommended that two separate accesses be performed: the first access setting the STP bit to stop the timer, followed by two NOP instructions (to provide a delay of at least eight CK2 periods), then the second access setting the ACK bit to clear the TIP flag.

PORTP Port P output control register. Contains bits to individually program the outputs from PWM Timers T4 — T7 that are presented on the Port P output pins. See also Section 9.6.

1						7	Word a	at 015	2						
T7 TFN 0		B	T7 OUT 0	T6 TFN 0	*	٠	T6 OUT 0	T5 TFN 0	*	*	T5 OUT 0	T4 TFN 0	*		T4 OUT 0
	Pin P3	Control			Pin P2	Control			Pin P1	Control			Pin P0	Control	
	Byte at 0153									Byte a	t 0152	ļ			

0 Indicates that the bit is cleared on Reset.

Unassigned bit. Should be considered indeterminate data.

OUT Programs the initial Port P output signal level. These bits become readonly once Toggle mode has been entered by setting the TFN bit.

OUT = 1 Places the corresponding Port P pin at logic 1.

OUT = 0 Places the corresponding Port P pin at logic 0.

TFN Specifies the functions of the port pins PO(T4OUT)—P3(T7OUT).

TFN = 1 Enables the corresponding OUT bit to be toggled on an underflow from its Timer. While TFN is set, the OUT bit is read-only to software.

TFN = 0 Disables toggling. The Port P pin continues to take its value from the OUT bit, which may be changed by software.

NOTE: An OUT bit should not be altered in the same access that sets the corresponding TFN bit to a one; doing so can affect the OUT bit unpredictably. Instead, set the OUT bit to the desired value first, then set the TFN bit to a one in a subsequent instruction. Clearing the TFN bit while changing the OUT bit in the same access is allowed, and performs both changes reliably.

16.5 TIMER SETUP AND CONFIGURATION

The following sequence of operations is recommended to initialize the timers and their I/O pins for an application:

1. Stop and reset the timers:

Stop timers T1—T3 and clear interrupt enables on T0—T3 by writing the value 4440 (hex) into the TMMODE Register.

Execute two NOP instructions, or any other appropriate instructions, to provide a delay of at least 8 cycles of the CK2 clock.

Clear all PND bits for timers T0—T3 by writing the value CCC8 (hex) into the TMMODE Register (setting all ACK bits while keeping the STP bits set and the TIE bits clear).

Stop timers T4—T7 and disable interrupts from them by writing 4444 (hex) into the PWMODE register.

Execute two NOP instructions, or any other appropriate instructions, to provide a delay of at least 8 cycles of the CK2 clock.

Clear all PND bits for timers T4—T7 by writing the value CCCC (hex) into the PWMODE register (setting all ACK bits while keeping the STP bits set and the TIE bits clear).

2. Configure Timer I/O Pins:

Table 16-1 shows the possible modes of operation for the B3 (T2IO) and B4 (T3IO) pins. For each of these bits that will be an output, write the desired initial values into the corresponding bits of the PORTB register. Set the DIRB and BFUN register bits 3 and 4 to select the desired modes of operation (ignoring the DIVBY register for now).

Table 16-2 shows the possible modes of operation for the pins B8 (TS0), B9 (TS1), B13 (TS2) and B14 (TS3). For each of these bits that will be an output, write the desired initial values into the corresponding bits of the PORTB register. Set the DIRB and BFUN register bits 8, 9, 13 and 14 appropriately to select the desired modes of operation.

Write the desired initial states into the T4OUT—T7OUT bits of the PORTP register, then set the desired T4TFN—T7TFN bits to enable toggling on timer underflows. (Note that the initial state of these pins is zero on reset.)

3. Configure and Set Up Timers:

Select the desired functions for registers I2CR/I3CR (R1/T1) by setting the T0CON register bit T0 ORG appropriately.

Select the desired mode for the I4CR register by setting the T0CON register bit T0 C4F appropriately.

Select the desired clock sources for Timers T2 and T3 by loading the upper byte of the DIVBY register appropriately.

Load all desired timers and their input registers with their initial values.

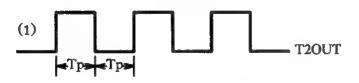
In the TMMODE and PWMODE registers, turn on the appropriate interrupt enable (TIE) bits. Also, before starting the timers, ensure that the TIMERS bit (bit 5) of the ENIR register is set and that the ENIR register bit GIE is also set. This is necessary in order for timer interrupts to be recognized.

4. Start the timers by clearing the appropriate STP bits in the TMMODE and PWMODE registers.

16.6 TIMER APPLICATIONS

The following examples illustrate the use of the timers. Generation of square wave frequencies, constant and variable duty cycle frequencies, triggered pulses and synchronous pulse patterns are requirements common to many applications. These examples show the timers being used to solve these generic problems.

SQUARE WAVE FREQUENCY GENERATION



GL-32-0-U

(1) User software:

Loads Tp into T2. (Tp = 1/2 of desired period)
Loads Tp into R2.
Starts Timer T2.

The output is a constant squarewave frequency. No further software attention is required.

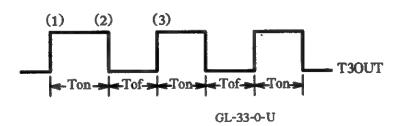
TABLE 16-1. FUNCTIONS OF PINS B3 (T2IO) AND B4 (T3IO)

MOD	E SELE	CTION	BEH	AVIOR OF P	IN, PORTB R	EGISTER BI	T AND ASS	OCIATED T	IMER
BFUN Bit	DIRB Bit	Timer Clock Source (DIVBY)	Pin Asserted as Output	PORTB Bit Action On Timer Underflow	Pin Action On Timer Underflow	Timer Action On Falling Edge	PORTB Bit Action On Software . Write	Pin Action On Software Write	Data Source On Software Read
0	0	Int.	no	none	none	none	load	none	pin
0	0	Ext	по	none	none	count	load	none	pin
0	1	Int.	yes	none	none	none	load	follows PORTB bit	PORTB register bit
0	1	Ext.	yes	none	none	count	load	follows PORTB bit	PORTB register bit
1	0	Int.	no	toggle	none	none	none	none	pin
1	. 0	Ext.	110	toggle	none	count	none	none	pin
1	1	înt.	yes	toggle	toggle	none	zone	noné	PORTB register bit
1	1	Ext.	yes	(timer stops)	(timer stops)	(timer stops)	none	none	PORTB register bit

TABLE 16-2. TS0-TS3 PIN FUNCTIONS

МО	DE		BEHAVIOR OF PIN AND PORTB REGISTER BIT									
BFUN	DIRB	PIN ASSERTED AS OUTPUT	PORTB BIT ACTION ON TIMER T2 UNDERFLOW	PIN ACTION ON TIMER T2 UNDERFLOW	PORTB BIT ACTION ON SOFTWARE WRITE	PIN ACTION ON SOFTWARE WRITE	DATA SOURCE ON SOFT WARE READ					
0	0	no	none	none	load	none	pin					
0	1	yes	none	none	load	follows PORTB bit	PORTB register bit					
1	0	no	togg le	none	none	none	pin					
1	1	yes	togg le	togg le	none	none	PORTB register bit					

CONSTANT AND VARIABLE DUTY CYCLE GENERATION



(1) User software:

Loads Ton into T3.

Loads Tof into R3.

Enables Timer T3 interrupt.

Sets pin B4 (T3OUT) high, and enables it to toggle.

Starts Timer T3.

(2) On Interrupt from T3:

Hardware toggles T3OUT and copies contents of R3 to

T3. User software loads Ton into R3.

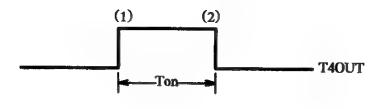
(3) On Interrupt from T3:

Hardware toggles T3OUT and copies contents of R3 to

T3. User software loads Tof into R3.

On each interrupt from T3 the user software alternately loads Ton and Tof into Register R3. The result is a constant duty cycle output. A constant frequency, variable duty cycle signal can be generated by changing the Ton and Tof values such that their sum remains constant.

SINGLE PULSE GENERATION



GL-34-0-U

(1) User software:

Loads Ton into T4.

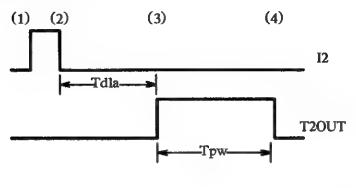
Enables Timer T4 interrupt. Sets output pin P0 (T4OUT). Enables T4OUT to toggle.

Starts Timer T4.

(2) On Timer T4 interrupt:

Hardware toggles T4OUT. User software stops T4.

EXTERNALLY TRIGGERED DELAYED PULSE GENERATION



GL-35-0-U

(1) User software:

Stops Timer T2 and resets pin B3 (T2OUT).

Loads Tdla into T2. Loads Tpw into R2. Enables T2 interrupt.

Enables interrupt on I2 falling edge (Trigger).

Sets up I2CR/R1 as a capture register.

Sets up I4CR to load continuously from Timer T0.

(2) On interrupt from I2:

User software subtracts contents of I2CR from I4CR, yielding the interrupt latency time. The Tdla value in T2 is adjusted based on this value. Software then starts T2.

(3) Upon Timer T2 interrupt:

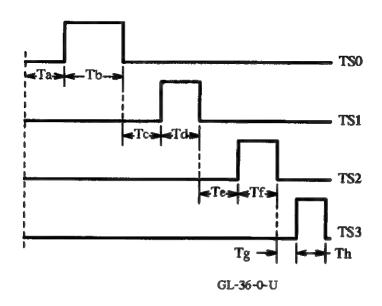
Hardware toggles T2OUT and copies contents of R2 into T2. User software notes that this event has occurred, but does nothing else.

(4) Upon Timer T2 interrupt:

Hardware toggles T2OUT. User software stops T2.

The result is a delayed pulse. Note that the Tdla value must be adjusted to include the Interrupt response time. By capturing the time of the trigger event in the I2CR register, and comparing it with the current contents of Timer T0, the variable interrupt service latency is determined and can be compensated for, keeping Tdla constant. The width of the pulse is inherently independent of interrupt service latency times, depending only on the values placed in T2 and R2.

SYNCHRONOUS PULSE GENERATION



(1) User software:

Loads Ta into T2.

Loads Tb into R2.

Enables Timer T2 interrupt.

Resets TS0-TS3.

Enables TSO to toggle. Others are disabled.

Starts Timer T2.

(2) Upon Timer T2 interrupt:

Hardware toggles TSO and copies contents of R2 into

T2.

User software loads Tc into R2.

(3) Upon Timer T2 interrupt:

Hardware toggles TSO and copies contents of R2 into

T2.

User software loads Td into R2, and enables only TS1

to toggle.

Steps (2) and (3) are repeated; user software loading a new delay value into R2 on each interrupt and enabling the next output on every second interrupt, until all four outputs have been pulsed in sequence.

Chapter 17

MICROWIRE/PLUS

17.1 INTRODUCTION

MICROWIRETM is a synchronous serial communication scheme, originally implemented on the COPSTM family of microcontrollers to minimize the number of connections (and thus the cost) required to communicate with peripherals. The implementation used in the HPC16083 is an enhancement of the COPS MICROWIRE interface, and is called MICROWIRE/PLUS. It remains compatible with all MICROWIRE peripherals.

Figure 17-1 shows a typical set of interconnections among MICROWIRE devices. Each device has a serial input pin (called SI or DI), a Serial Output pin (SO or DO) and a Serial Clock pin (SK or CLK), which is presented by one processor (the "Master") and input by all other processors and peripherals (the "Slaves"). The HPC has the capability of operating as either a Master or a Slave. An additional signal, Chip Select, is required by most peripheral chips, and can be provided from any port bit on the Master HPC.

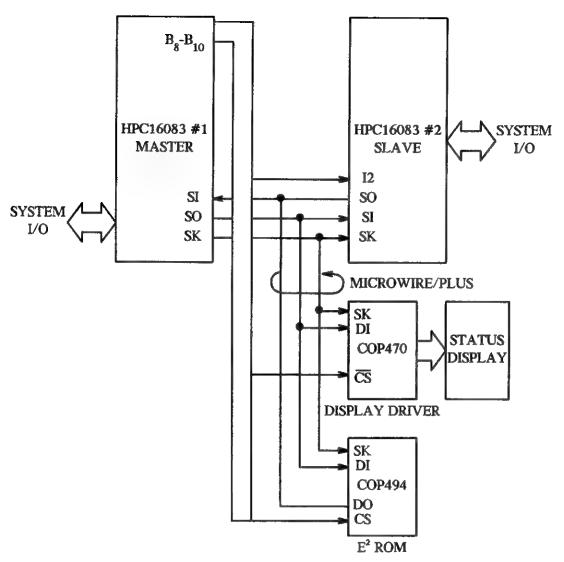
The MICROWIRE/PLUS implementation consists of the three-wire interface (SI, SO and SK), working in conjunction with an eight-bit input/output shift register (SIO). Figure 17-2 shows a block diagram of the MICROWIRE/PLUS interface circuit.

Figure 17-3 illustrates the sequence followed in a MICROWIRE transfer. The transfer is triggered whenever the program writes to the SIO register. This clears a flag called uWDONE (bit 0 of register IRPD, see Section 15.5), which enables the SIO register to shift. If the HPC is in Master Mode, this also automatically causes the SIO register to be shifted left eight times, and eight SK pulses are presented on the SK pin. If the HPC is in Slave Mode, shifting is enabled but does not actually occur until the SK pulses are received from the Master device. On each rising edge of SK, the MICROWIRE/PLUS interface latches a value from the SI pin. On the falling edge of SK the SIO register shifts, presenting the next bit on the SO pin and placing the value previously latched from SI into its least-significant bit. When all eight shifts have been performed, the uWDONE flag is set. Note that data is transferred most-significant bit first.

17.2 REGISTERS

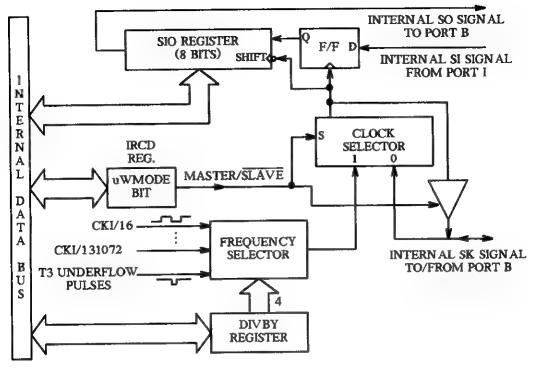
The MICROWIRE/PLUS interface uses four registers: the SIO register, which serves as the shift register for transfers, and fields of the IRCD (Interrupt Condition), IRPD (Interrupt Pending) and DIVBY (Divide-By) registers.

The SIO register occupies a single byte at address 00D6.



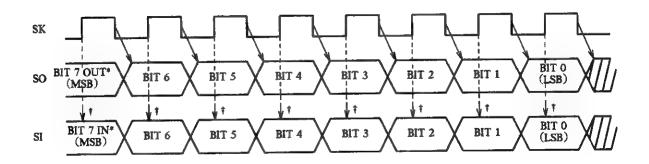
GL-37-0-U

Figure 17-1. Example of a MICROWIRE Application



GL-38-0-U

Figure 17-2. MICROWIRE/PLUS Interface Block Diagram



- * This bit becomes valid immediately when the transmitting device loads its SIO register.
- † Arrows indicate points at which SI is sampled.

GL-39-0-U

Figure 17-3. MICROWIRE/PLUS Transfer

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			SIO — By	te at 00D6			

NOTE: Some instructions (e.g., X and ADD) perform both a read and a write access to their destination operands. MICROWIRE transfers only occur on the write access, so read-modify-write accesses to the SIO register have the effect of transferring the value finally written. Care should be taken not to write to SIO while it is shifting data; this can cause an undefined value to be written. In Slave mode, the SIO register does not begin shifting until SK clock pulses are received from the Master, therefore, any number of accesses (read or writes) may be performed to the SIO as long as they are completed before the Master starts sending SK pulses.

In the IRPD register, bit 0 is the uWDONE flag, which is reset on starting a transfer and is set on completion. Although it occupies a bit in the IRPD register, the uWDONE flag does not trigger an interrupt. It is read-only to software.

EI	UART	TIMERS	INT4	INT3	INT2	NMI	uWDONE
XR	XR	XR	XC	XC	XC	XR	XR
			IRPD — By	yte at 00D2			

In the IRCD register, bit 1 is the uWMODE bit, which combines with certain Port B register settings to determine whether the HPC operates its interface as a MICROWIRE Master or as a Slave (see Section 17.1).

**	** O	0	I4 POL 0	I3 POL 0	12 POL 0	uWMODE 0	RDY/HLD 0
	<u></u>		IRCD — B	yte at 00D4			

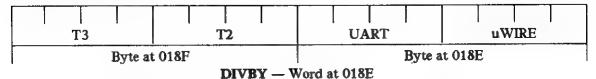
O Cleared on Reset.

X State indeterminate on Reset.

R Read-only.

C Clearable by writing zero, writing a one does nothing.

The DIVBY register, in bits 0—3, serves to program the rate of the SK clock generated by the HPC in Master Mode. SK can come from any of 14 pre-scaled versions of the crystal frequency (CKI), or from Timer T3, as listed below.



BIT 3	BIT 2	BIT 1	BIT 0	MICROWIRE CLOCK
0	0	0	0	Not Allowed
0	0	0	1	Defined by Timer T3 underflows *
0	0	1	0	CKI / 16
0	0	1	1	CKI / 32
0	1	0	0	CKI / 64
0	1	0	1	CKI / 128
0	1	1	0	CKI / 256
0	1	1	1	CKI / 512
1	0	0	0	CKI / 1024
1	0	0	1	CKI / 2048
1	0	1	0	CKI / 4096
1	0	1	1	CKI / 8192
1	1	0	0	CKI / 16384
1	1	0	1	CKI / 32768
1	1	1	0	CKI / 65536
1	1	1	1	CKI / 131072

* Each underflow of T3 corresponds to one cycle on SK; placing a value n in T3 and R3 performs division by n+1 on the T3 input clock. The T3-derived SK waveform is not symmetrical in general, and is therefore incompatible with most existing MICROWIRE peripherals. It does, however, support communication among HPCs. The low times between SK pulses are fixed at eight cycles of the CKI clock, with the signal high for the remainder of each cycle. This asymmetry remains even if T3 divides an off-chip, symmetrical clock source by 1.

17.3 INITIALIZATION

The SO output signal is the alternate function assigned to Port B pin B5. To enable this function, bit 5 of the BFUN register and bit 5 of the DIRB register must both be set to ones.

The SI input is automatically taken from the Port I pin I5. No action is required to configure this pin.

The SK clocking function is the alternate function assigned to Port B pin B6. The configuration of Port B pin B6, however, involves three configuration bits instead of two. These are bit 6 of

the BFUN register, bit 6 of the DIRB register and the uWMODE bit of the IRCD register. Not all combinations of these bits are legal, as shown in the table below.

BFUN Bit 6	DIRB Bit 6	uWMODE (IRCD Bit 1)	Function Selected for Pin B6
0	0	0	SK clock input (MICROWIRE Slave Mode). This mode also allows the pin to be used as a Port B input.
0	0	1	CAUTION
0	1	0	Port Output.
0	1	1	CAUTION
1	0	0	Invalid Selection. (Slave Mode with invalid SK clock input.)
1	0	1	Port B input only. (MICROWIRE interface in Master Mode but with no SK output.)
1	1	0	Invalid Selection. (Slave Mode with invalid SK clock input.)
1	1	1	SK clock output (MICROWIRE Master Mode).

The note CAUTION indicates mode selections that should not be entered even momentarily, as they bring two on-chip buffers into conflict. Damage to the HPC chip itself is unlikely, but these selections can result in excessive power consumption and/or unpredictable system operation.

All three bits above are cleared automatically on reset.

The uWDONE flag is not cleared on reset, and may indicate a pending transfer. Given this and the constraints above, the following procedure is recommended for initializing the MICROWIRE/PLUS interface:

- 1. Set bit 5 in the BFUN and DIRB registers. This configures the B5 pin as SO.
- 2. Set BFUN register bit 6. This selects the SK function for pin B6.
- 3. Select CKI/16 as the timebase for the interface by placing the binary pattern "0010" into the least-significant four bits of the DIVBY register.
- 4. Set the uWMODE bit (bit 1 of the IRCD register). This places the interface in Master mode. If the uWDONE bit was not already set, the interface will now shift and set it.

5. Wait until the uWDONE flag is set to 1, then continue initializing the interface. If Master Mode is desired, set DIRB bit 6 in order to make SK an output, and select the desired timebase in the least-significant four bits of the DIVBY register. If Slave Mode is desired, clear first the uWMODE bit, then bit 6 of the BFUN register.

Note that the SO and SK pins float on reset, as do all other Port B pins. It is recommended that the SK pin be connected to a pull-down resistor to ensure that no spurious pulses are received by the MICROWIRE peripherals.

		· ·

Chapter 18

UART

18.1 INTRODUCTION

The HPC16083 is equipped with a software programmable on-chip UART. The features of the UART are as follows.

18.2 ASSOCIATED I/O PINS

Data is transmitted on the TDX pin and received on the RDX pin. TDX is the alternate function assigned to Port B pin B0; it is selected by setting bit 0 of the BFUN register and bit 0 of the DIRB register. RDX is an inherent function of Port I pin I6, requiring no setup. The normal state of the RDX pin is the MARK level (high) except when a character is being received.

The baud rate clock for the UART can be generated on-chip, or it can be taken from an external source. Port B pin B2 has as its alternate function the clocking function CKX, selected by setting bit 2 of the BFUN register. The CKX pin can be either an input or an output, as determined by bit 2 of the DIRB register. As an input, it accepts a clock signal which may be selected to drive the transmitter and/or the receiver. As an output, it presents the clock signal being used by the UART.

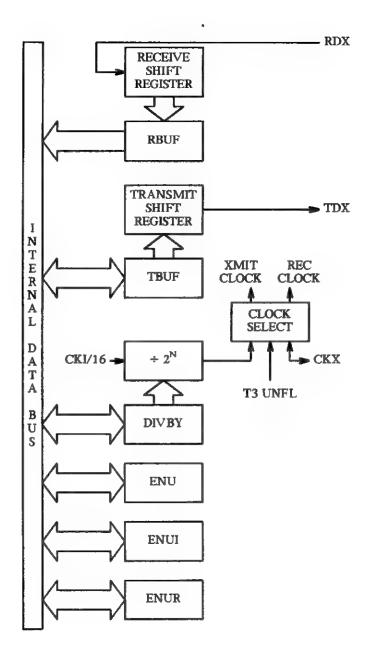
18.3 UART OPERATION

Figure 18-1 shows a block diagram of the UART. The UART consists of a Transmitter section and a Receiver section. It has five addressable registers: TBUF, RBUF, ENU, ENUR and ENUL. These registers are eight bits wide. There are also two non-addressable shift registers, called RSFT and TSFT.

The registers ENU, ENUR and ENUI contain control and status bits for the UART.

The TSFT and TBUF registers double-buffer data for transmission. While TSFT is shifting out the current character on the TDX pin, the TBUF register may be loaded by software with the next byte to be transmitted. When TSFT finishes transmitting the current character the contents of TBUF are transferred to the TSFT register and the Transmit Buffer Empty Flag (TBMT, in the ENU register) is set. The TBMT flag is automatically reset by the UART when software loads a new character into the TBUF register. TBUF is a read/write register.

The RSFT and RBUF registers double-buffer data being received. The UART receiver continually monitors the signal on the RDX pin for a low-going edge to detect the beginning of a start bit. Upon sensing this edge, it waits for half of a bit time and samples again. If the RDX pin is still low, the receiver considers this to be a valid START bit, and the remaining bits in the character frame are each sampled a single time, at the mid-bit position. Serial data input on the RDX pin is shifted into the RSFT register. Upon receiving the complete character, the contents of the RSFT register are copied into the RBUF register and the Receive Buffer Full Flag



GL-40-0-U

Figure 18-1. UART Block Diagram

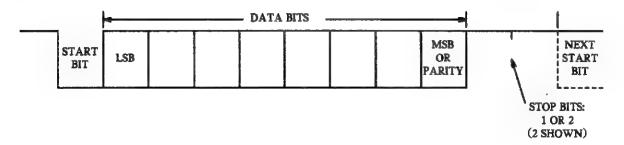
(RBFL) is set. The Receive Buffer Full Flag is automatically reset when software reads the character from the RBUF register. RBUF is a read-only register.

TBMT and RBFL are read-only bits. The user cannot set or reset them by writing to the ENU register.

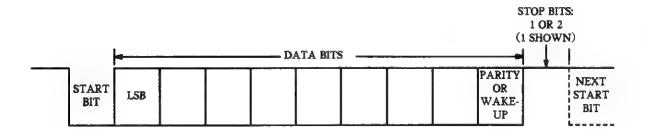
NOTE: The UART is automatically reset whenever the HALT or IDLE powerdown mode is entered. The transmitter output freezes at its current state, and all control registers assume their reset states.

18.4 FRAME FORMATS SUPPORTED

The UART supports two serial frame formats, illustrated in Figure 18-2. The format is selected using the bit 80R9 in the ENU register.



8-BIT MODE



9-BIT MODE

GL-41-0-U

Figure 18-2. Character Frame Formats

The first format for data transmission (80R9 = 0) consists of one Start bit, eight Data bits and one or two Stop bits (as selected in the 2STP bit of the ENUI register). This format supports transfer of eight-bit characters without parity, or seven-bit characters with parity. In applications using parity, the most significant bit of the TBUF and RBUF registers is the parity bit, which must be generated and verified by software. Upon receiving a character, the flag RBIT9 in the ENUR register also holds this bit.

The second format for transmission (80R9 = 1) consists of one Start bit, nine Data bits and one or two Stop bits. This supports transmission of eight-bit characters with parity, and also supports the UART "Wake-Up" feature (see Section 18.8). When operating in this format, all eight bits of TBUF and RBUF are used for data. The ninth data bit (or parity bit) is transferred using two bits in the control registers, called XBIT9 and RBIT9. XBIT9, located in the ENU register, is transmitted as the parity bit or as a ninth data bit. RBIT9, located in the ENUR register, holds this bit from received characters. Figure 18-3 illustrates the sources and destinations of UART data.

The frame formats for the receiver differ from the transmitter in the number of Stop bits required. The receiver only requires one Stop bit in a frame, regardless of the setting of the 2STP bit.

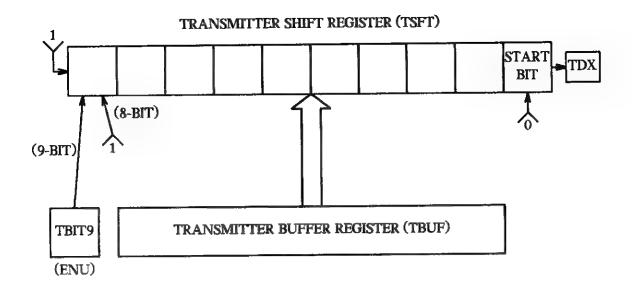
18.5 ERROR FLAGS

Error conditions detected by the Receiver section are brought to the software's attention by setting two error flags in the ENUR register. The flag bit FE (Framing Error) is set when the receiver fails to see a valid Stop bit at the end of the frame, and the flag bit DOE (Data Overrun Error) is set if a new character is received into RBUF while it is still full (i.e., before software has read the previous character). The UART continues to update the RBUF register with each character received even when it detects an overrun or framing error.

The FE and DOE bits are read-only to software. They are reset only when software reads them as ones, or on a UART reset (applying RESET to the HPC or entering HALT or IDLE power-down mode). This is to ensure that there is no gap between the time software checks for errors and the time an action is taken that clears error flags. There are two considerations, then, in accessing the ENUR register:

- 1. Never read the ENUR register without checking the FE and DOE bits in the image read (or saving the image for later checking).
- 2. Never perform a read-modify-write operation (e.g., a bit operation or an exclusive OR) on the ENUR register directly, since this reads (and clears) the DOE and FE bits without saving them anywhere to test them.

There is no hardware to allow automatic generation or checking of parity. These functions must be done in software.



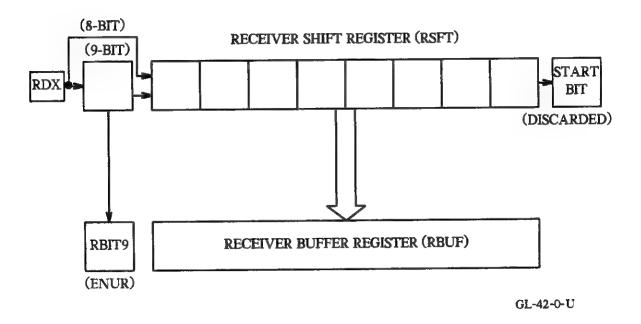
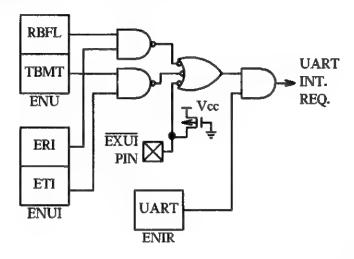


Figure 18-3. Data Flow Between TSFT/RSFT and Other Registers

18.6 INTERRUPTS

The UART is capable of generating interrupt requests. Interrupts from the Transmitter and the Receiver may be independently enabled and disabled, using the ETI and ERI bits in the ENUI register. If both interrupts have been enabled the interrupt service routine must determine the interrupt source by examining the RBFL and TBMT flags in the ENU register.

Figure 18-4 illustrates how the UART interrupt request is derived.



GL-43-0-U

Figure 18-4. UART Interrupt Generation

The interrupt from the Transmitter is set pending, and remains pending, as long as both the TBMT and ETI bits are set. To remove this interrupt, software must either clear the ETI bit or write to the TBUF register (thus clearing the TBMT bit). UART interrupts can also be disabled by clearing the UART bit in the ENIR register (see Section 18.9).

The interrupt from the Receiver is set pending, and remains pending, as long as both the RBFL and ERI bits are set. To remove this interrupt, software must either clear the ERI bit or read from the RBUF register (thus clearing the RBFL bit). UART interrupts can also be disabled by clearing the UART bit in the ENIR register (see Section 18.9).

The UART interrupt can also be triggered using the pin \overline{EXUL} This pin is a level-sensitive input that requests an interrupt as long as it is low. The \overline{EXUL} pin is provided primarily to support HPC applications that do not need the UART, but which require an additional interrupt input. Interrupt requests from \overline{EXUL} can be removed either by pulling the pin high or by clearing the UART bit in the ENIR register (see Section 15.5). An on-chip pull-up device holds the pin high (inactive) if no external connection is made to it.

18.7 CLOCKING

The clock inputs to the Transmitter and Receiver sections of the UART can be independently selected to come either from an external source on the CKX pin or from a source selected in the DIVBY register. This selection is performed using the XTCLK and XRCLK bits in the ENUI register. The input frequency to the UART must be 16 times the required baud rate.

The control bit XTCLK selects the UART Transmitter clock. Setting it high, in conjunction with the appropriate Port B mode settings (Section 9.3) causes the Transmitter clock to be obtained from an external source on the CKX pin. Resetting it allows the DIVBY register to define the clock. Similarly, the control bit XRCLK allows the user to program the source for the UART Receiver section.

Bits 4—7 of the DIVBY register select the clock source (when it does not come from the CKX pin), as shown in Table 18-1. In addition to prescaled versions of the HPC's crystal frequency, a clock may be derived from underflows of Timer T3 (one cycle per underflow). Note however that if this clock signal is selected for output on the CKX pin (as explained in Section 18.2) the clock signal presented is asymmetrical; the low time of the signal is fixed at eight cycles of the CKI clock.

It is possible to operate the Transmitter and the Receiver sections of the UART at different baud rates. This can be done by deriving the clock for one of the sections from DIVBY and using the signal on the CKX pin for the other section. By specifying Timer T3 as the clock source in the DIVBY register, and clocking T3 from its external input (T3IO, on pin B4), it is even possible to run the two UART sections from independent external frequencies. Placing zero into both the T3 and R3 registers performs a division by one on the T3IO frequency, in effect passing it directly through to the UART. Placing another value n into T3 and R3 has the effect of prescaling the external clock by the value n+1.

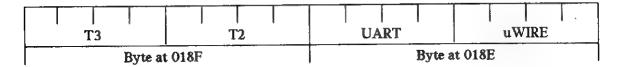
18.8 WAKE-UP MODE

The UART Receiver section supports an alternate mode of operation, referred to as Wake-Up Mode. This mode of operation is selected by setting the WAKEUP bit in the ENUR register. The data format for transmission must also be selected as having nine Data bits and either one or two Stop bits.

The Wake-Up mode of operation is intended for use in networking the HPC16083 with other processors. Typically in such environments, the "messages" consist of a "header" (a device address) followed by the "body" (actual data intended for a specific device). This mode of operation enables the environment to contain a number of devices. The header determines which device will receive the data in the body of the message. The Wake-Up mode supports a scheme in which headers (8-bit addresses) are flagged by having the ninth bit of the data field set to a 1. The body of the message consists of 8-bit data bytes, with the ninth bit of the data field always reset to 0.

While in Wake-Up Mode, the HPC UART monitors the communication flow, but ignores all characters until an address character is received. Upon receiving an address character, the UART signals that the character is ready by setting the RBFL flag, which in turn interrupts the processor if UART Receiver interrupts are enabled. The WAKEUP bit is also cleared automatically at this point, so that data characters as well as address characters are recognized.

TABLE 18-1. UART CLOCK SOURCES FROM DIVBY REGISTER



DIVBY: Word at 018E

BIT 7	BIT 6	BIT 5	BIT 4	BAUD CLOCK	BAUD RATE	BAUD RATE
}				(x16 Clock)	9.83 Mhz	16.88 Mhz
					Crystal	Crystal
0	0	0	0	<-	- Not Allowed -	->
0	0	0	1	<-Defined	by Timer T3 und	lerflow>
0	0	1	0	CKI / 16	38400	65536
0	0	1	1	CKI / 32	19200	32768
0	1	0	0	CKI / 64	9600	16384
0	1	0	1	CKI / 128	4800	8192
0	1	1	0	CKI / 256	2400	4096
0	1	1	1	CKI / 512	1200	2048
1	0	0	0	CKI / 1024	600	1024
1	0	0	1	CKI / 2048	300	512
1	0	1	0	CKI / 4096	150	256
1	0	1	1	CKI / 8192	75	128
1	1	0	0	CKI / 16384	38	64
1	1	0	1	CKI / 32768	19	32
1	1	1	0	CKI / 65536	9.4	16
1	1	1	1	CKI / 131072	4.7	8

Software examines the contents of the RBUF register and responds by deciding either to accept the subsequent data stream (by leaving the WAKEUP bit reset) or to wait until the next address character is seen (by setting the WAKEUP bit again).

Operation of the UART Transmitter is not affected by selection of Wake-Up Mode. The value of the ninth bit to be transmitted is programmed by setting the ENU register bit XBIT9 appropriately. The value of the ninth bit received is obtained by reading ENUR register bit RBIT9.

18.9 UART CONTROL REGISTERS

The operation of the UART is programmed through three registers: ENU, ENUR and ENUL The function of the individual bits in these registers is as follows:

ENU -- UART Control and Status

*	*	XBIT9 0	8OR9 0	*	*	RBFL OR	TBMT 1R
			Byte a	t 0120			

ENUR — UART Receive Control and Status

DOE 0D	FE 0D	*	*	RBIT9 OR	WAKEUP 0	*	*
			Byte a	t 0128			

ENUI - UART Interrupt and Clock Source Register

2STP 0	*	*	*	XRCLK 0	XTCLK 0	ERI 0	ETI 0
			Byte a	t 0122		-	

- Unassigned bit. Should be considered indeterminate data.
- O Bit is cleared on reset.
- 1 Bit is set to a one on reset.
- R Bit is read-only: it cannot be written by software.
- D Bit is destructively read: when read by software as a one, it is cleared automatically. Writing to the bit does not affect its state.

NOTE: The bits in these registers assume their initial states when either a Reset occurs or the HPC enters Halt or Idle power-down mode.

18.9.1 Detailed Description of UART Register Bits

FE Flags a Framing Error, as defined in Section 18.5.

FE = 1 Indicates the occurrence of a Framing Error.

FE = 0 Indicates no Framing Error has been detected since the last time the ENUR register was read.

DOE Flags a Data Overrun Error, as defined in Section 18.5. DOE = 1Indicates the occurrence of a Data Overrun Error. DOE = 0Indicates no Data Overrun Error has been detected since the last time the ENUR register was read. RBIT9 Contains the ninth data bit received when the UART is operating with nine data bits per frame. XBIT9 Programs the ninth data bit for transmission when the UART is operating with nine data bits per frame. 8OR9 Selects the character frame format. 80R9 = 0The frame contains eight data bits. 80R9 = 1The frame contains nine data bits. WAKEUP Wake-Up Mode is enabled while this bit is set. This bit is cleared automatically on receiving a character with data bit 9 set. **RBFL** This bit is set when the UART has received a complete character and has copied it into the RBUF register. It is automatically reset when the HPC16083 reads the character from RBUF. This bit is set when the UART transfers a byte of data from the TBUF register into TBMT the TSFT register for transmission. It is automatically reset when software writes into the TBUF register. **ERI** This bit enables/disables interrupts from the Receiver section. ERI = 0Interrupts from the receiver are disabled. ERI = 1Interrupts from the receiver are enabled. ETI This bit enables/disables interrupts from the Transmitter section. ETI = 0Interrupts from the transmitter are disabled. ETI = 1Interrupts from the transmitter are enabled. This bit selects the clock source for the Receiver section. XRCLK XRCLK = 0The clock source is selected through the DIVBY register. XRCLK = 1Signal on CKX pin is used as the clock. This bit selects the clock source for the Transmitter section. XTCLK The clock source is selected through the DIVBY register. XTCLK = 0XTCLK = 1Signal on CKX pin is used as the clock. This bit programs the number of Stop bits to be transmitted. 2STP 2STP = 0One Stop bit transmitted. Two Stop bits transmitted. 2STP = 1

Chapter 19

UNIVERSAL PERIPHERAL INTERFACE (UPI)

19.1 INTRODUCTION

The Universal Peripheral Interface (UPI) feature allows the HPC16083 to be used as a peripheral device, directed through a parallel bus interface by another processor system. Since the other processor initiates all data exchanges it will be referred to here as the Host Processor.

The UPI interface consists of a Data Bus (UDB: 8 or 16 bits wide), two data strobe inputs (URD and UWR), two handshake outputs (RDRDY and WRRDY) and one Address input (UA0).

In UPI Mode, Port A becomes the UPI Data Bus. Port B pins B10, B11 and B15 become the UA0 input and the $\overline{\text{RDRDY}}$ and $\overline{\text{WRRDY}}$ outputs, respectively. The Port I pins I2 and I3 become the $\overline{\text{URD}}$ and $\overline{\text{UWR}}$ inputs, respectively, and retain their ability to interrupt the HPC.

NOTE: The pin assignment used for UPI Mode requires the use of pins that would otherwise be available for bus control signals in the Expanded and ROMless modes of operation. The on-chip UPI interface therefore cannot be used unless the HPC16083 is in Single-Chip Normal Mode. In Single-Chip ROMless Mode the HPC supports re-creation of the UPI function off-chip.

19.2 INTERNAL ORGANIZATION

Figure 19-1 illustrates the internal organization of the UPI interface. The interface contains a control register (UPIC), and makes use of the two registers PORTA and DIRA as buffer registers. The PORTA register (at address 00F0) is used as the output buffer register, which holds data to be read by the Host Processor. In this role it is referred to as the OBUF register. The DIRA register (at address 00E0) is used as the input buffer register, which receives data from the Host Processor. It is referred to as the IBUF register in this role. In 8-Bit UPI mode, only the low-order byte of each register is used for the UPI interface; the high-order byte retains its bit-programmable I/O functions.

The UPI Data Bus is implemented on the Port A pins, and can be configured to be either eight or sixteen bits wide. When eight-bit UPI Mode is selected, only pins A0—A7 are used as the UPI bus, and pins A8—A15 remain usable for bit-programmable I/O. The bus is normally floating, and is not driven by the HPC except while the host is reading from it.

The inputs $\overline{\text{URD}}$ and $\overline{\text{UWR}}$ are the Read and Write strobes, labelled according to the conventions of the Host Processor: $\overline{\text{URD}}$ is pulsed to read from the HPC, and $\overline{\text{UWR}}$ is pulsed to write to it. The actions performed are shown in Figure 19-2. When the host writes a value to the HPC, it is latched into the IBUF register on the rising edge of the $\overline{\text{UWR}}$ strobe and remains there to be inspected by HPC software. When HPC software writes a value into the OBUF register, it is held there for the host. When the host pulses the $\overline{\text{URD}}$ pin low, data is presented on the bus. This data may come from either the OBUF register or the UPIC register, as selected by the host using the UAO pin.

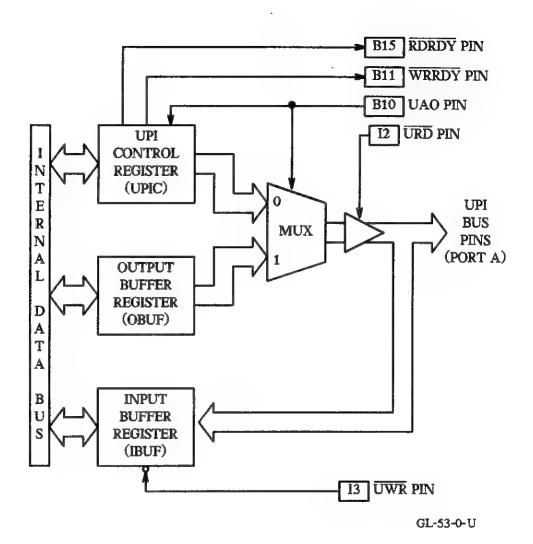


Figure 19-1. UPI Block Diagram

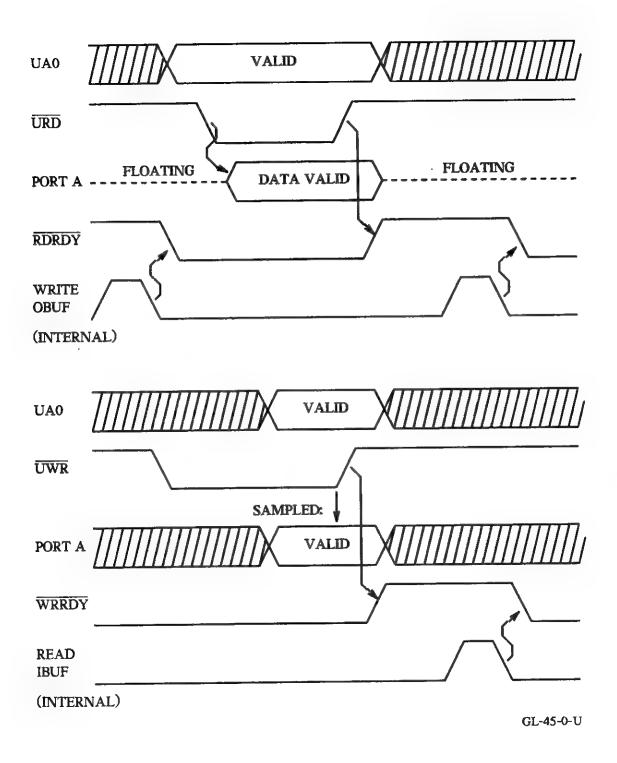


Figure 19-2. UPI Timing Diagrams

The UPI interface does not include Chip Select logic to allow selective addressing of the HPC16083 by the Host Processor. This must be done by externally qualifying $\overline{\text{URD}}$ and $\overline{\text{UWR}}$ with Chip Select logic so that they are generated only if the HPC has also been selected.

19.3 UPI CONTROL REGISTER

The UPI Control register (UPIC) allows UPI Mode to be entered. It also contains status information to allow polled as well as interrupt-driven handshake operation. The UPIC register can be read by the Host Processor as well as by HPC software. The following is a functional description of the individual bits:

Q	Q	Q	8OR16	UPIEN	LA0	RDRDY	WRRDY
XR	XR	XR	X	0	XR	1R	OR
			UPIC: By	te at 00E6	Ť		

- O Bit is cleared on reset.
- 1 Bit is set to a "1" on reset.
- X Bit state is indeterminate on reset.
- R Bit is read-only to software.

BIT FUNCTION

WRRDY

Status Bit. This bit is an active low output from the UPI interface, indicating when the Host Processor may write data into the Input Buffer register (IBUF). It is normally low (active), and is set high when the Host Processor writes data into IBUF. It is set low again when the HPC reads from IBUF. This bit may also be presented on Port B pin B11 as a status signal for the Host Processor (see Section 19.5).

RDRDY

Status Bit. This bit is an active low output from the UPI interface, indicating when the host may read from the output buffer register (OBUF). It is normally high (inactive), and is set low (active) when the HPC writes data into the output buffer. It returns high when the host reads the output buffer. This bit may also be presented on Port B pin B15 as a status signal for the Host Processor (see Section 19.5).

LAO

This bit latches the Address Input pin UAO on the rising edge of the UWR strobe. It is in effect an extra data bit written by the Host Processor, and can be used, for example, to distinguish between commands and data sent to the HPC.

UPIEN Enables/Disables UPI mode:

UPIEN = 1 UPI mode is enabled.

UPIEN = 0 UPI mode is disabled.

80R16 Selects either 8-Bit or 16-Bit UPI Mode:

80R16 = 1 The UPI data bus is eight bits wide.

80R16 = 0 The UPI data bus is sixteen bits wide.

Q These bit positions present the corresponding bits from the OBUF (PORTA) register. When accessed through the UPIC register, these bits are read-only.

When the UPIC register is read as a 16-bit value, by either the HPC or the Host Processor, all of the extra bits come from the OBUF register. It is therefore possible to mix data and status information in the same transfer. Care should be taken, however, because accessing the UPIC register does not affect the WRRDY and RDRDY flags.

19.4 PIN FUNCTION DETAILS

Figure 19-2 illustrates the timing sequences that occur during UPI Read and Write operations.

19.4.1 UPI Write Operation

The Host Processor writes to the HPC by presenting data on the bus and pulsing the \overline{UWR} line low. Four events occur on the rising (trailing) edge of \overline{UWR} :

- 1. The contents of the data bus are latched in the IBUF register.
- 2. The UAO pin (B10) is sampled and the value is placed into the LAO bit of the UPIC register. The UAO pin serves no other purpose in UPI Write transfers. Software in the HPC can use the LAO bit in the UPIC register, for example, to distinguish between commands and data.
- 3. An interrupt is generated to the HPC, if the UWR pin (I3) is conditioned to trigger its interrupt on rising edges.
- 4. The WRRDY bit in the UPIC register, normally low, goes high, indicating that the IBUF register is full. If Port B pin B11 is configured as the WRRDY output (Section 19.5), it also goes high at this time. The WRRDY bit goes low again when the HPC reads the IBUF register.

19.4.2 UPI Read Operation

A UPI Read is generally performed in a sequence of events that starts with HPC software writing a value into the OBUF register. The RDRDY bit in the UPIC register, normally high, goes low, indicating that the OBUF register is full. If the pin B15 is enabled as the RDRDY pin, it also goes low at this time. The Host Processor, seeing that data is available for it, initiates a UPI Read cycle by performing the following steps:

- The Host Processor sets the UAO pin appropriately, then activates the URD pin (to its low state). The HPC hardware responds according to the value seen on the UAO pin:
 - UA0 = 0 The contents of the UPIC register are placed on the data bus.
 - UA0 = 1 The contents of the OBUF register are placed on the data bus.
- 2. The Host Processor then inputs the data from the bus and deactivates the URD pin. If the host has read from the OBUF register, the RDRDY bit in the UPIC register goes inactive (high) on the rising edge of URD. Reading from the UPIC register has no effect on the RDRDY bit. If Port B pin B15 is configured as the RDRDY output (Section 19.5), it follows the state of the RDRDY bit. An interrupt is also generated to the HPC, if the URD pin (12) is conditioned to trigger its interrupt on rising edges.

The UPIC register is only five bits wide. When either the Host Processor or the HPC accesses it, the remaining portion of the data comes from the OBUF register.

The logic controlling RDRDY and WRRDY does not support selective access to the individual bytes of the OBUF and IBUF registers. An access to either half of one of these registers is interpreted as an access to the entire register.

Use of the pins WRRDY and RDRDY is optional in a UPI application. Polled handshaking can be performed by the host processor without them, freeing these pins for use as bit-programmable I/O instead. See Section 19.5 for further details.

* The name "UAO" implies that the pin should be connected to the least-significant bit of the Host Processor's address bus. This, however, is correct only if the Host Processor's data bus is one byte wide. UAO should not be connected to any address line that specifies only the alignment of data on the bus. For example, the least-significant address bit from a host that addresses bytes on a 16-bit bus should not be used.

19.5 HANDSHAKING

The WRRDY and RDRDY bits of the UPIC register, and their corresponding output pins, are provided to allow handshaking between the Host Processor and the HPC. Three basic schemes can be implemented to synchronize Host Processor accesses:

No handshaking signals at all. In this, the simplest scheme, the Host Processor reads
the UPIC register and tests the WRRDY and RDRDY bits to determine when the
HPC is ready to transfer data. See Figure 19-3. In some applications, the UAO pin is

not necessary; for example, in a situation where the Host Processor writes only data and reads only the UPIC register. In such situations, UAO can be programmed as an output, thus providing its own pull-up or pull-down. See Section 19.6.

- 2. Interrupting the Host Processor using the WRRDY and RDRDY pins. Some interfacing examples are shown in Figures 19-4 and 19-5. In this scheme, the Host Processor is interrupted when the WRRDY or RDRDY pin goes low. The interrupts can be either level-sensitive or edge-sensitive, but falling-edge sensitive interrupts may be more convenient in some systems due to the fact that the WRRDY signal is normally active.
- 3. Holding the Host Processor in WAIT states whenever:
 - a. the Host Processor attempts to read from the HPC and the RDRDY pin is high (inactive), or
 - b. the Host Processor attempts to write to the HPC and the WRRDY pin is high (inactive).

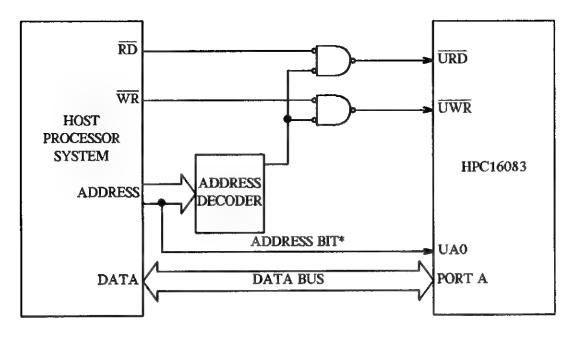
This scheme yields the highest throughput in communication, but may be more expensive. To support this, the HPC allows the \overline{URD} pin to be activated while \overline{RDRDY} is high (inactive), as long as the rising edge does not appear until \overline{RDRDY} goes low again. The \overline{UWR} pin, however, should be gated so that it is asserted to the HPC only when \overline{WRRDY} is active. This is because IBUF is a fall-through register, and must not be changed until the HPC has read from it. Figure 19-6 illustrates the connections required to use this scheme. Figures 19-7 and 19-8 shows the timing sequences involved in such a system for reading and writing.

19.6 UPI MODE SETUP

UPI Mode is entered by setting the UPIEN bit in the UPIC register. In UPI Mode, Port A automatically becomes the UPI Data Bus. If the UPIC register bit 80R16 is reset to "0", all 16 bits of Port A are used; otherwise only pins A0—A7 are used. The Port I pins I2 and I3 become the URD and UWR inputs, respectively.

Port B pins B10, B11 and B15 may be used as the UA0 input and the RDRDY and WRRDY outputs, respectively. To set these up for UPI operation, some initialization must also be performed on Port B. To select the proper modes for the WRRDY pin, bit 11 of both the DIRB and BFUN registers must be set. Likewise, the RDRDY pin must be configured for its function by setting bit 15 of both the DIRB and BFUN registers.

Selection of the UAO pin function is automatic; bit 10 of the BFUN register need not be set to select it. However, the UAO pin must also be designated as an input pin by clearing bit 10 of the DIRB register. If the function of the UAO pin is not needed in an application, the pin can be designated as an output instead. This causes the UAO value to be taken, in effect, from the PORTB data register, saving an external pull-up or pull-down device.



GL-46-0-U

* The name "UAO" implies that the pin should be connected to the least-significant bit of the Host Processor's address bus. This, however, is correct only if the Host Processor's data bus is one byte wide. UAO should not be connected to any address line that specifies only the alignment of data on the bus. For example, the least-significant address bit from a host that addresses bytes on a 16-bit bus should not be used.

Figure 19-3. Simple UPI Application

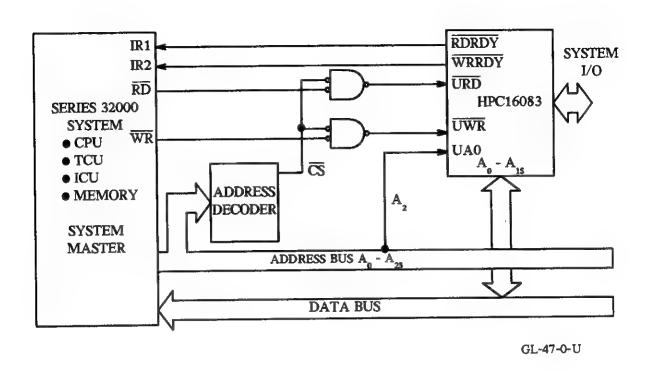


Figure 19-4. Interrupt-Driven UPI Application: UPI Interface to Series 32000 System

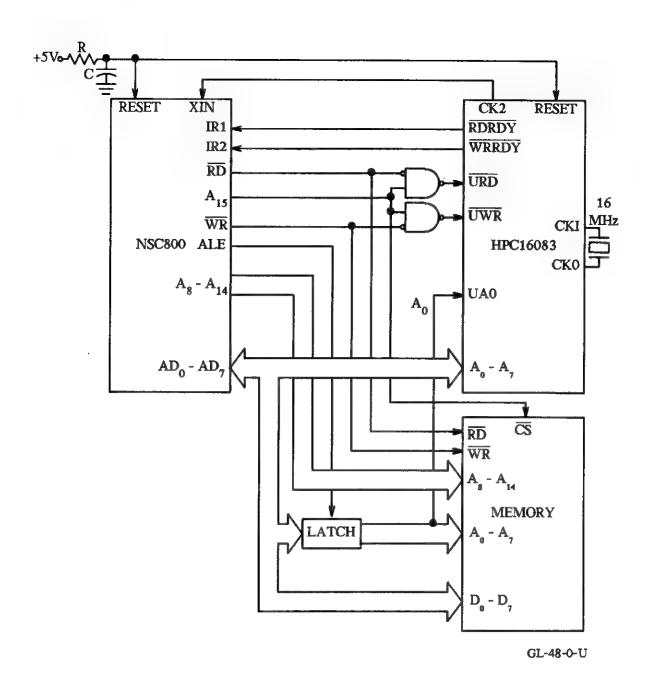


Figure 19-5. Interrupt-Driven UPI Application: UPI Interface to NSC800

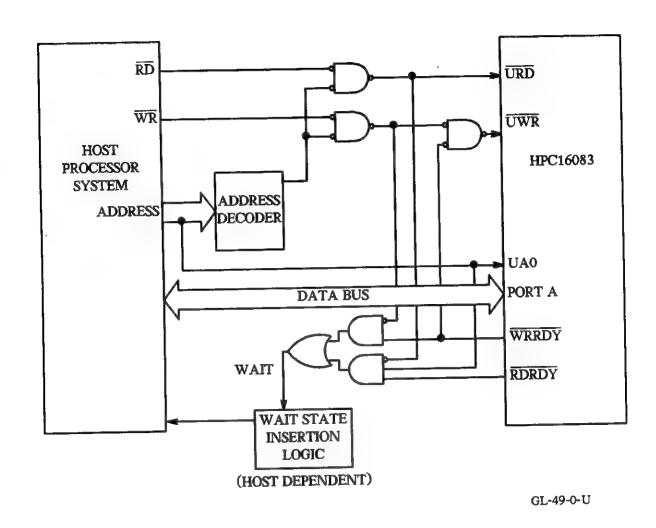
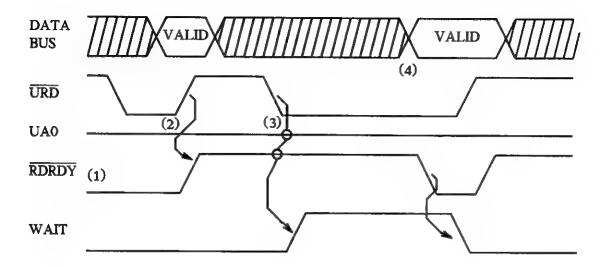


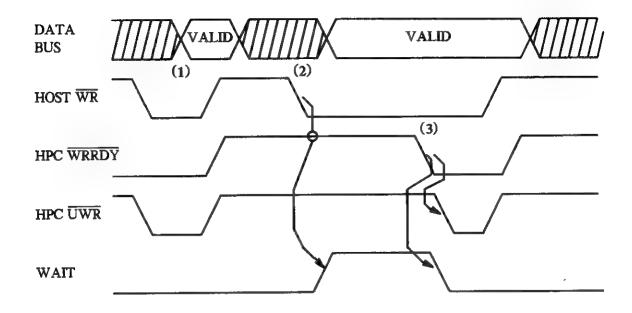
Figure 19-6. UPI Application Inserting WAIT States



- (1) Assumes the OBUF register has already been filled by the HPC.
- (2) Host reads from OBUF, emptying it. RDRDY goes high.
- (3) Host attempts to read from empty OBUF. It is held in Wait states.
- (4) HPC loads OBUF. New data becomes valid on bus, then RDRDY goes low. Host Wait states are terminated.

GL-50-0-U

Figure 19-7. Host Wait State Insertion Sequence of Events (Reading)



- (1) OBUF register is empty; this write transfer is performed at full speed.
- (2) Host processor starts second write to HPC; IBUF still full. UWR must not be activated, since it would overwrite IBUF. Host is held in wait states.
- (3) HPC empties IBUF. WRRDY goes active, activating UWR. Host wait states are terminated.

GL-51-0-U

Figure 19-8. Host Wait State Insertion Sequence of Events (Writing)

		^
		_

Appendix A REGISTERS AND MEMORY MAP

A.1 HPC PROGRAMMING MODEL

A	Accumulator	W(00C8)
В	Primary Address Register	W(00CC)
x	Secondary Address Register	W(OOCE)
K	Boundary Register	W(00CA)
PC	Program Counter	W(00C6)
SP	Stack Pointer	W(00C4)
PSW	Status & Carry	M(00C0)

A.2 HPC16083 MEMORY MAP

FFFF:FFF0	Interrupt Vectors in On-Chip ROM	
FFEF:FFD0	JSRP Vectors in On-Chip ROM	1
FFCF:FFCE : E001:E000	On-Chip ROM	USER MEMORY
DFFF:DFFE : 0201:0200	External Expansion Memory	
01FF:01FE : 01C1:01C0	On-Chip RAM	User RAM
0195:0194	Watchdog Register	Watchdog Logic
0192 0191:0190 018F:018E 018D:018C 018B:018A 0189:0188 0187:0186 0185:0184 0183:0182 0181:0180 015F:015E 015C	TOCON Register TMMODE Register DIVBY Register T3 Timer R3 Register T2 Timer R2 Register I2CR Register/R1 I3CR Register/T1 I4CR Register EICR Register EICN Register	Timer Block TO:T3
0153:0152 0151:0150 014F:014E 014D:014C 014B:014A 0149:0148 0147:0146 0145:0144 0143:0142 0141:0140 0128 0126 0124 0122	Port P Register PWMODE Register R7 Register T7 Timer R6 Register T6 Timer R5 Register T5 Timer R4 Register T4 Timer ENUR Register TBUF Register RBUF Register ENUI Register	Timer Block T4:T7

0104	Port D Input Register	PORT D
00F5:00F4 00F3:00F2 00F1:00F0	BFUN Register DIR B Register DIR A Register / IBUF	PORTS A & B CONTROL
00E6	UPIC register	UPI CONTROL
00E3:00E2 00E1:00E0	Port B Port A / OBUF	PORTS A & B
00DE 00DD:00DC 00D8 00D6 00D4 00D2 00D0	Microcode ROM Dump Halt Enable Register Port I Input Register SIO Register IRCD Register IRPD Register ENIR Register	PORT I CONTROL & INTERRUPT CONTROL REGISTERS
00CF:00CE 00CD:00CC 00CB:00CA 00C9:00C8 00C7:00C6 00C5:00C4 00C3:00C2 00C0	X Register B Register K Register A Register PC Register SP Register (reserved) PSW Register	HPC CORE REGISTERS
00BF:00BE : 0001:0000	On-Chip RAM	USER RAM

A.3 HPC16083 REGISTER BIT MAPS

	MSB							LSB
PSW	* X	CGIE X R	C X	EA 0	WAIT1 0	WAITO 0	HLT/IDL 0	EHI 0
BFUNL	HLDA	SK	SO	TM3 I/O	TM2 I/O	CKX	*	TDX
DECINE	0	0	0	0	0	0	0	0
BFUNH	RDRDY	TS3	TS2	*	WRRDY	*	TS1	TSO
DI CIVII	0	0	0	0	0	0	0	0
ENIR	EI	UART	TIMERS	INT4	INT3	INT2	*	GLOBAL
TO ARRA	X	X	X	X	X	X	Х	0
IRCD	*	*	*	I4 POL	I3 POL	I2 POL	uW MODE	
IKCD	0	0	0	0	0	0	0	0
IRPD	EI	UART	TIMERS	INT4	INT3	INT2	NMI	uW DONE
KID	XR	XR	XR	XC	XC	XC	XR	XR
ENU	*	*	XBIT9	8OR9	*	*	RBFL	TBMT
LIVO			0	0			OR	1 R
ENUR	DOE	FE	*	*	RBIT9	WAKEUI	*	*
2.101	OD	0D		<u> </u>	O R	0		
ENUI	2STP	*	*	*	XRCLK	XTCLK	ERI	ETI
	0	<u> </u>			0	0	0	0
UPIC	Q	Q	Q	80R16	UPIEN	LAO	RDRDY	WRRDY
	XR	XR	XR		0	R	1R	OR

DVBYH	TIMER T3 DIVIDE BY			TIMER T2 DIVIDE BY				
DVBYL	-	UART DI	VIDE BY		4 -	uWIRE D	IVIDE BY	
TOCON	*	*	*	*	TO ORG	T0 C4F X	*	*
TMMDL	T1 ACK 0W	T1 STP X	T1 TIP XR	T1 TIE X	TO ACK OW	*	TO TIP XR	TO TIE
TMMDH	T3 ACK 0W	T3 STP X	T3 TIP XR	T3 TIE	T2 ACK 0W	T2 STP X	T2 TIP XR	T2 TIE X
PWMDL	T5 ACK	T5 STP X	T5 TIP XR	T5 TIE	T4 ACK 0W	T4 STP X	T4 TIP XR	T4 TIE
PWMDH	T7 ACK	T7 STP X	T7 TIP	T7 TIE	T6 ACK 0W	T6 STP X	T6 TIP XR	T6 TIE
PRTPL	T5 TFN	*	*	T5 OUT	T4 TFN	*	*	T4 OUT
	0 T7 TFN	*	*	T7 OUT	T6 TFN	*	*	T6 OUT
PRTPH	0	*	1	0	0	ACK	MODE	POL
EICON	*	*	*	*	*	1W	0	0

NOTE: * Bit is unassigned, and should be considered indeterminate data.

RBIT9 Bit Name
0 R

- 0 → Cleared on Reset
- 1 → Set on Reset
- X -> State indeterminate on Reset
- R → Read-Only
- W → Write-Only
 (Reads back as its initial state.)
- C → Clearable by writing zero.
 Writing a one does nothing.
- D → Destructive Read-Only Reading clears any set bits.
- Q → These bit positons present the corresponding bits from the OBUF (PORTA) register.
 When accessed through the UPIC register, these bits are read-only.

Appendix B

INSTRUCTION SET SUMMARY

Arithme	etic instructions				
ADD	Add	MA ← MA+MemI	C ← carry		
ADC	Add with carry	MA ← MA+MemI+C	$C \leftarrow carry$		
ADDS	Add short imm8	$A \leftarrow A + imm8$	C ← carry		
DADC	Decimal add with carry	MA(decimal) ← MA+MemI+C	C ← carry		
SUBC	Subtract with carry	MA ← MA-MemI+C	C ← carry		
DSUBC	Decimal subtract w/carry	MA(decimal) ← MA-MemI+C	$C \leftarrow carry$		
MULT	Multiply (unsigned)	MA & X ← MA*Memī, K ←0	C ← 0		
DIV	Divide (unsigned)	MA ← MA/Meml, X ←rem., K ←0	C ← 0		
DIVD	Divide double word	MA ← (MA & X)/MemI,			
		$X \leftarrow \text{rem., } K, C \leftarrow 0$	$C \leftarrow 1$ if ovf. or divide by 0		
IFEQ	If equal	Compare MA & MemI, Do next if eq	ual		
IFGT	If greater than	Compare MA & Meml, Do next if M	A > Meml		
AND	Logical and	MA ← MA and MemI			
OR	Logical or	MA ← MA or MemI			
XOR	Logical exclusive-or	MA ← MA xor Memi			
Memor	y Modify instructions				
INC	Increment	Mem ← Mem + 1			
DECSZ	Decrement, skip if 0	Mem ← Mem - 1, Skip next if Mem	= 0		
Bit inst	ructions				
SBIT	Set bit	Mem.bit ← 1	bit = 0 to 7 immediate,		
			or dynamic: M(B).X		
RBIT	Reset bit	Mem.bit ← 0			
IFBIT	If bit	If Mem.bit is true, do next instr.			
Memor	y Transfer instructions				
LD	Load	MA ← Memi			
ST	Store to memory	A → Mem			
X	Exchange	A ←→ Mem			
PUSH	Push memory to stack	$W(SP) \leftarrow W, SP \leftarrow SP+2$			
POP	Pop stack to memory	$W \leftarrow W(SP), SP \leftarrow SP-2$			
LDS	Load A, incr/decr B,	$A \leftarrow Mem(B), B \leftarrow B \pm 1(or 2),$			
	Skip on condition	Skip next if B greater/less than K			
XS	Exchange,incr/decr B,	$A \longleftrightarrow Mem(B), B \longleftrightarrow B \pm 1(or 2)$			
	Skip on condition	Skip next if B greater/less than K			
LD	Load, incr/decr X	$A \leftarrow \text{Mem}(X), X \leftarrow X \pm 1 \text{ (or 2)}$			
X	Exchange, incr/decr X	$XA \longleftrightarrow Mem(X), X \leftarrow X \pm 1 (or 2)$			

LDB	Load B immediate	B ← imm
LD K	Load K immediate	K ← imm
LD X	Load X immediate	X ← imm
LD BK	Load B and K immediate	$B \leftarrow \text{imm}, K \leftarrow \text{imm}$
Accumulate	or and C instructions	
CLR A	Clear A	A ← 0
INC A	Increment A	A ← A+1
DEC A	Decrement A	A ← A-1
COMP A	Complement A	A ← one's complement of A
SWAP A	Swap nibbles of A	A1512 ← A118 ← A74 ←→ A30
RRC A	Rotate A right thru C	$C \rightarrow A15 \rightarrow \rightarrow A0 \rightarrow C$
RLC A	Rotate A left thru C	$C \leftarrow A15 \leftarrow \leftarrow A0 \leftarrow C$
SHR A	Shift A right	$0 \to A15 \to \to A0 \to C$
SHL A	Shift A left	C ← A15 ← ← A0 ← 0
SC	Set C	C ← 1
RC	Reset C	C ← 0
IFC '	IF C	Do next if C = 1
IFNC	IF not C	Do next if C = 0
Control Tra	ansfer instructions	
JSRP†	Jump subroutine from table	$W(SP) \leftarrow PC, SP \leftarrow SP+2$
		$PC \leftarrow W(table\#)$
JSR	Jump subroutine relative	W(SP) \leftarrow PC, SP \leftarrow SP+2, PC \leftarrow PC+# (# is +1025 to -1023)
JSRL	Jump subroutine long	$W(SP) \leftarrow PC, SP \leftarrow SP+2, PC \leftarrow PC+4$
JP	Jump relative short	$PC \leftarrow PC + \# (\# is + 32 \text{ to } -31)$
JMP	Jump relative	$PC \leftarrow PC + \# (\# \text{ is } +257 \text{ to } -255)$
JMPL	Jump relative long	PC ← PC+#
JID	Jump indirect at PC + A	$PC \leftarrow PC+1+A + M(PC+1+A)$
JIDW	Jump indirect at PC + A	$PC \leftarrow PC+1+A + W(PC+1+A)$
NOP	No Operation (= JP next)	PC ← PC+1
RET	Return	$PC \leftarrow W(SP), SP \leftarrow SP-2$
RETSK	Return then skip next	$PC \leftarrow W(SP)$, $SP \leftarrow SP-2$, & skip
RETI	Return from interrupt	PC ← W(SP), SP ← SP-2, interrupts reenabled

[†] JSRP is not an assembler mnemonic. Use "JSR", after declaring the subroutine name with the ".SPT" assembler directive.

SYMBOLS:

A	is the 16-bit Accumulator
В	is the 16-bit B register
X	is the 16-bit X register
K	is the 16-bit K register
PC	is the 16-bit Program Counter
SP	is the 16-bit Stack Pointer
C	is the Carry flag
imm	is 8-bit or 16-bit immediate data
imm8	is 8-bit immediate data
Mem	is an 8-bit byte or 16-bit word of memory
W	is a 16-bit word of memory
MA	is the Accumulator or direct memory, 8- or 16-bit (destination)
MemI	is 8-bit or 16-bit memory, or 8-bit or 16-bit immediate data (source)

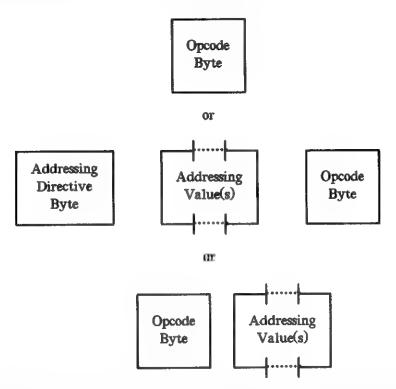
		`
	,	
		`

Appendix C

INSTRUCTION SET ENCODING, LENGTH AND TIMING

C.1 INSTRUCTION STRUCTURE

In memory, an instruction has one of the following forms:



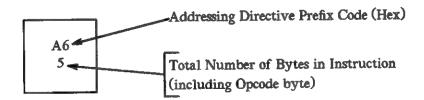
In most instructions, the one-byte opcode can stand by itself, representing the addressing mode combination "A, [B].B" or "A, [B].W".

To express the other addressing modes, an Addressing Directive byte is placed first, which redefines the addressing mode. This byte is followed by the necessary addressing values (addresses, displacements and/or immediate values), and the Opcode byte appears last. When there are two operands requiring addressing information (as in the Direct-Direct and Immediate-Direct modes), the addressing information for the Source (rightmost) operand appears first. When an addressing value is two bytes in length, the most-significant byte appears first.

Some special purpose instructions and some very commonly used instruction types use the third form, in which a limited range of addressing modes is encoded directly within the Opcode byte. In these cases, the Opcode byte is followed by the required addressing value(s), no Addressing Directive being required.

C.2 ADDRESSING DIRECTIVE PREFIXES AND RESULTING INSTRUCTION LENGTHS

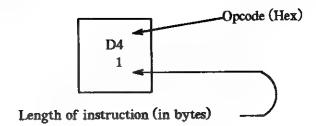
		BYTE S	OURCE	WORD	SOURCE		
MODE	CASE	BASE	NON- BASE	BASE	NON- BASE		
Direct-Direct	Base Dest.	80 4	84 5	A0 4	A4 5		
	Non-Base Dest.	81 5	85 6	A1 5	A5 6		
ImmedDirect	Base Dest.	8	32 4		86 5		
	Non-Base Dest.		33 5	87 6			
Indexed	8-bit Disp.		,	A2 4			
	16-bit Disp.	A6 5					
Direct	Base	96 3					
i i	Non-Base]	B6 4	1		
Indirect				AD 3			
X-Indirect [X]				8F 2	· · · · · · · · · · · · · · · · · · ·		



NOTE: "Base" means that the operand is within the Basepage (the first 256 bytes of the addressing space). Such operands require only an 8-bit addressing field in the instruction to address them. The Basepage contains RAM and the most frequently used registers (memory mapped).

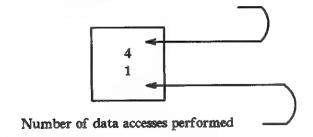
C.3 INSTRUCTION TABLE CONTENTS

Opcode/Length
Table Entries



Cycles (Minimum Execution Time) in units of CK2 clock cycles

Cycles/Accesses
Table Entries



C.4 INSTRUCTION EXECUTION TIME

Texec = C + (Wi * L) + (Wd * A)

C = Cycles (minimum execution time), from Cycles/Accesses Table.

Wi = Number of WAIT states imposed on instruction fetches.

Wd = Number of WAIT states imposed on data references.

L = Instruction length in bytes, from Opcode/Length Table.

A = Number of data accesses, from Cycles/Accesses Table.

The result is in units of the CK2 clock, which is half the frequency of the crystal input.

NOTE: Wi and Wd are based upon where the instruction opcodes and data bytes are located. For instance, Wi would be zero if opcodes were being fetched out of on-chip RAM, and would be the number of programmed Wait states if opcodes were being fetched out of on-chip ROM.

C.5 INTERRUPT PROCESSING TIME

Tintr = (11 + 3Wi + Wd) cycles of the CK2 clock.

C.6 SKIP PROCESSING TIME

Tskip = (4 + Wi) cycles of the CK2 clock.

C.7 OPCODES AND LENGTHS (TWO-ADDRESS INSTRUCTIONS)

	OPERAND			TO	ACCUMULAT	OR				MORY	
MNEM ONIC	SIZE	[B]	(x)	DIRECT	ENDIRECT	INDE XED	IMMED.	SOURCE RYTE	→ DIRECT SOURCE WORD	SOURCE RYTE	→ DIRECT SOURCE WORD
ADC	Byte	C8	C8	C8	C8	C8	E8	C8	-	C8 *	-
	Word	E8	E8	E8 #	E8	E8	E8	E8 *	E8 *	E8	E8
ADD	Byte	D8	De	D8	D8	D8	B8 3	D8		D8	-
	Word	F8	F8	F8	F8	F8	B8 3	F0	F8	F8	F8
AND	Byte	D9 1	D9	D9	D9	109	99	D9 *	-	D9	-
	Word	F9	F9	P9	F9	P9	B9 3	F9 *	F9	F9	£9
DADC	Byte	C9	C9	C9	C9	C9	E9	C9	-	C9	-
	Word	E9	E9	E9	E9	Ey	E9	E9	E9	E9	E9
DIV	Byte	DF	DF	DF	DF	DF	9F 2	DF	-	DF	-
	Word	FF	FF	FF	FF	FF #	BF 3	FF	FF .	FF	FF +
DIVD	Byte	CF	CF	CF	CF	CF #	CF #	CF	-	CF	-
	Word	EF	EF	EF	EF	EF	EF #	EF .	EF	EF +	EF.
DSUBC	Byte	CA	CA	CA	CA	CA	EA #	CA	-	CA	-
	Word	EA 1	EA *	EA	EA	EA	EA #	EA	EA .	EA	EA ‡
IFEQ	Byte	DC	DC	DC	DC	DC	9C	DC	-	DC	-
	Word	FC	FC	FC	FC	FC	BC 3	FC	FC	FC	FC
IFGT	Byte	DD	DD	DD	DD	DD	9D 2	DD		DD *	-
	Word	FD	FD	FD	FD	FD	BD 3	FD	FD #	FD	FD .
LD	Byte	C	D4	88, 88	88	88	90	8C, 88	-	97, 8B	-
	Word	E4	F4	A8, A8	A8	A8	B0 3	AB #	AC, AB	AB	87, AB
MULT	Byte	DE	DE	DE	DE	DE	9E	DE	-	DE	-
	Word	FE	FE	FE	FE	FE	BE 3	FE	FE t	FE #	FE +
OR	Byte	DA	DA	DA	DA *	DA *	9A 2	DA *	-	DA *	-
	Word	FA	FA	FA	FA	FA *	BA 3	FA	FA	FA	FA ¢
SUBC	Byte	СВ	CB	СВ	CB	CB	EB et	CB +	-	CB	-
	Word	EB	EB	EB	EB	EB	EB #†	EB	EB #	EB *	EB +
XOR	Byte	DB	DB	DB	DB	DB	9B 2	DB		DB	-
	Word	FB	FB	FB	FB	FB	BB 3	FB	FB *	FB	FB *

NOTE: When two opcodes are given, the first applies only if all operands are in the Base Page (the first 256 bytes of memory).

- indicates that the opcode requires an Addressing Directive prefix. The prefix determines the length of the instruction.
- † indicates that this form (Immediate to Accumulator) does not exist as a separate opcode. The assembler substitutes the Immediate-to-Memory form, using the fact that the Accumulator is memory-mapped at address OOC8 Hex.

C.8 CYCLES AND ACCESSES (TWO-ADDRESS INSTRUCTIONS)

TO ACCUMULATOR				то	ACCUMULA	TOR				MORY	
MN	EMONIC	[B]	[x]	DIRECT	INDIRECT	INDEXED	IMMED.	DIRECT SOURCE BASE	→ DIRECT SOURCE NON-BASE	IMMED. SOURCE BYTE	→ DIRECT SOURCE WORD
ADC	8-bit Field (Base)	4	7	8 1/0	10 2/1	13 2/1	11* 2/0	13 3/0	15 3/1	11 2/0	13 2/0
	16-bit Field (Non-Base)	1	Ĺ	10 1	2/1	15 2/1	13* 2/0	15 3/2	17 3	13 2	15 2
ADD	8-bit Field (Base)	4	7	8 1/0	10 2/1	13 2/1	0	13 3/0	15 3/1	11 2/0	13 2/0
	16-bit Field (Non-Base)		,	10 1	6/1	15 2/1	6	15 3/2	17 3	13 2	15 2
AND	8-bit Field (Base)	4	7	8 1/0	10 2/1	12 2/1	4 0	13 3/0	15 3/1	11 2/0	13 2/0
	16-bit Field (Non-Base)	_	·	10 1	2/1	14 2/1	6	15 3/2	17	13 2	15 2
DADC	8-bit Field (Base)	6	9	10	12 2/1	15 2/1	13* 2/0	15 3/0	17 3/1	13 2/0	15 2/0
	16-bit Field (Non-Base)			12 1	57.	17 2/1	15* 2/0	17 3/2	19 3	15 2	17 2
DIV	8-bit Field (Base)	58 1	61 1	62 1/0	64 2/1	67 2/1	58 0	68 4/0†	70 4/1‡	66 3/0†	68 3/0‡
	16-bit Field (Non-Base)			64 1		69 2/1	6 0 0	70 4/3†	72 41	68 3†	70 3‡
DI∨D	8-bit Field (Base)	58 1	61	62 1/0	64 2/1	67 2/1	66 3/0‡	68 4/0‡	70 4/1‡	66 3/0†	68 3/0†
	16-bit Field (Non-Base)		-	64 1	2/1	69 2/1	68 3/01	70 4/3‡	72 41	68 3†	70 3†
DSUBC	8-bit Field (Base)	5	8	9 1/0	11 2/1	14 2/1	14° 3/0†	16 4/0‡	18 4/1†	3/0†	16 3/0†
	16-bit Field (Non-Base)			11	***	16 2/1	16* 3/0†	18 4/3‡	20 4†	16 3†	18 3†
IFEQ	8-bit Field (Base)	6	9	10 1/0	12 2/1	14 2/1	6	14 2/0	16 2/1	12 1/0	14 1/0
	16-bit Field (Non-Base)		_	12 1		16 2/1	8	16 2/1	18 2	14	16 1
IFGT	8-bit Field (Base)	5	8	1/0	11 2/1	13 2/1	5	15 3/0†	17 3/1‡	13 2/0†	15 2/0‡
	16-bit Field (Non-Base)			11 1		15 2/1	7	17 3/2†	19 3†	15 2†	17 2‡
LD	8-bit Field (Base)	4	4	6 1/0	10 2/1	13 2/1	0	10 2/0	16 2/1	1/0	14 1/0
	16-bit Field (Non-Base)		•	10 1		15 2/1	6	16 2/1	18 2	10	16
MULT	8-bit Field (Base)	58 1	61 1	62 1/0	64 2/1	67 2/1	58	68 4/0‡	70 4/1†	66 3/0†	68 3/0†
	16-bit Field (Non-Base)			64 1		69 2/1	60	70 4/31	72 4‡	68 3†	70 3†
OR	8-bit Field (Base)	4	7	8 1/0	10 2/1	12 2/1	0	13 3/0	15 3/1	2/0	13 2/0
	16 bit Field (Non Base)	^		10 1		14 2/1	6 0	15 3/2	17	13 2	15 2

		TO ACCUMULATOR						TO MEMORY DIRECT - DIRECT IMMED DIRECT			
MN	EMONIC	[B.]	[X]	DIRECT	INDIRECT	INDEXED	IMMED.	SOURCE	SOURCE NON-RASE	SOURCE BYTE	SOURCE
SUBC	8-bit Field (Base)	4	7	8 1/0	10	13 2/1	13* 3/0‡	15 4/0†	17 4/1‡	13 3/0‡	15 3/0‡
	16-bit Field (Non-Base)	1	1	10 1	2/1	15 2/1	15* 3/0‡	17 4/3†	19 41	15 3†	17 3‡
XOR	8-bit Field (Base)	4	7	8 1/0	10	12 2/1	4 0	13 3/0	15 3/1	11 2/0	13 2/0
	16-bit Field (Non-Base)	1	1	10	2/1	14 2/1	6	15. 3/2	17 3	13 2	15 2

NOTE: When two numbers appear for the number of Data Accesses, e.g., "2/0", only the second number of accesses is susceptible to WAIT states. Others are accesses to Base Page information, and do not receive WAIT states unless they access one of the registers associated with Port A, Port B or the UPI Port (addresses 00E0-00FF). In general, then, only the second number should be used for calculating instruction execution times.

- * indicates that this form (Immediate to Accumulator) does not exist as a separate opcode. The assembler substitutes the Immediate-to-Memory form, using the fact that the Accumulator is memory-mapped at address 00C8 (Hex). The numbers in the table reflect the result of the substitution.
- † indicates that this form of the given instruction reads twice from the destination operand; hence the extra access.

C.9 OPCODES AND LENGTHS (ONE-ADDRESS INSTRUCTIONS)

MNEMONIC	OPERAND SIZE	[B]	[x]	DIRI	ECT_	INDIRECT	INDEXED	X, [B].B
DECSZ	Byte	8A† *	8A *	8A, 2	8A *	8A *	8A *	-
	Word	AA†	AA *	AA, 2	AA *	AA *	AA *	-
IFBIT	<u> </u>	10-17 1	10-17	10-		10-17	10-17	3A 1
INC	Byte	89† *	89	89, 2	89	89 *	89 *	-
	Word	A9† *	A9 *	A9, 2	A9 *	A9 *	A9 *	-
RBIT		18-1F 1	18-1F *		-1F *	18-1F *	18-1F *	38
SBIT		08-0F 1	08-0F *	1	-OF	08-0F *	08-0F *	39 1
ST	Byte	C6	D6 1	8B, 2	8B *	8B *	8B *	-
	Word	E6 1	F6 1	AB, 2	AB *	AB *	AB *	-
x	Byte	C5	D5 1	8E, 2	8E *	8E *	8E *	ah
	Word	E5	F5	AE, 2	AE *	AE *	AE *	-

NOTE: When two opcodes are given, the first applies only if the operand is in the Basepage (the first 256 bytes of memory).

When a range of opcodes is given, e.g., "18-1F", these are the forms of a bit instruction that reference, respectively, bits 0 through 7 of the byte referenced by the addressing mode.

- * indicates that the opcode requires an Addressing Directive prefix. The prefix determines the length of the instruction.
- † indicates that this form (B-Register Indirect) does not exist as a separate opcode. The assembler substitutes the Indirect form, using the fact that the B Register is memory-mapped at address OOCC Hex.

C.10 CYCLES AND ACCESSES (ONE-ADDRESS INSTRUCTIONS)

MN	EMONIC	[B]	[x]	DIRECT	INDIRECT	INDEXED	X,[B].B	
DECSZ	8-bit Field (Base)	12†	9	8 2/0	12	15 3/2		
	16-bit Field (Non-Base)	3/2	2	12 2	3/2	17 3/2	-	
IFBIT	8-bit Field (Base)	6	9	10 1/0	12	15 2/1	13	
	16-bit Field (Non-Base)	1	1	12 1	2/1	17 2/1	1	
INC	8-bit Field (Base)	11†	11† 8 2/0 11 3/2 2 11 2	I .	14 3/2			
	16-bit Field (Non-Base)	3/2		1	3/2	16 3/2	-	
RBIT	8-bit Field (Base)	5	8	9 2/0	11	14 3/2	12	
	16-bit Field (Non-Base)	2	2	11 2	3/2	16 3/2	2	
SBIT	8-bit Field (Base)	5	8	9 2/0	11	14 3/2	12	
	16-bit Field (Non-Base)	2	2	11 2	3/2	16 3/2	2	
ST	8-bit Field (Base)	4 4 1	1	7 1/0	11	14 2/1		
	16-bit Field (Non-Base)		11 1	2/1	16 2/1			
X	8-bit Field (Base)	5	5	7 2/0	11	14 3/2		
	16-bit Field (Non-Base)	2	2	11 2	3/2	16 3/2	-	

NOTE: When two numbers appear for the number of Data Accesses, e.g., "2/0", only the second number of accesses is susceptible to WAIT states. Others are accesses to Basepage information, and do not receive WAIT states unless they access one of the registers associated with Port A, Port B or the UPI Port (addresses 00E0-00FF). In general, then, only the second number should be used for calculating instruction execution times.

† indicates that this form (B-Register Indirect) does not exist as a separate opcode. The assembler substitutes the Indirect form, using the fact that the B Register is memory-mapped at address OOCC Hex.

C.11 OPCODES AND LENGTHS (SPECIALIZED INSTRUCTIONS)

IMME	DIATE TO REGISTER	
	IMMI	EDIATE
MNEMONIC	вуте	WORD
LD B,#bval	92 2	B2 3
LD X,#xval	93 2	B3 3
LD K,#kval	91 2	B1
LD BK,#bval,#kval	8D 3	A7 5
ADDS A,#value	98 2	-

AUTO-INCREMENT/AUTO-DECREMENT

X REGISTER					
MNEMO	INIC	вуте	WORD		
LD A,	[X+]	D0 1	F0 1		
	[X-]	D2 1	F2 1		
Х А,	[X+]	D1 1	F1 1		
	[X-]	D3 1	F3 1		

	B REGISTER WITH CONDITIONAL SKIP					
MNEMO	DNIC	вуте	WORD			
LDS A,	[B+]	C0 1	E0 1			
	[B-]	C2 1	E2 1			
XS A,	[B+]	C1 1	E1 1			
	[B-]	C3 1	E3 1			

STACK POI	NTER
Mnemonic	Direct, Base, Word
РОР	3F 2
PUSH	AF 2

INSTRUCTIONS WITH INHERENT OPERANDS	
CLR A 00 1 1 INC A 04 1	_
INC A 04 1	
INC A 04	
1	
DEC 4	
DEC A 05	
1	
COMP A 01	
1	
SWAP A 3B	
1	
RRC A D7	
1	
RLC A F7	
1	
SHR A C7	
1	
SHL A E7	
11	
IFC 07	
1	
IFNC 06	
1	
SC 02	
1	
RC 03	
1	

NOTROCITO	NS THAT TRANSFER CONTR
JSRP	20-2F†
	1
ISR	00110ddd dddddddd
	2 †
ISRL	B5
	3
IP	01dddddd †
	1
MP	1001010d dddddddd
	2 †
MPL	B4
	3
ID	CC
	1
IDW	EC
	1
RET	3C
-	1
RETI	3E
	1
RETSK	3D
	1
NOP	40
	1

These instructions have a range of opcodes. The JSRP instruction references the table entry given by the least-significant 4 bits of the opcode. The JSR, JP and JMP instructions contain an embedded PC-relative displacement, and are shown in binary. Each d represents one bit of the displacement.

C.12 CYCLES AND ACCESSES (SPECIALIZED INSTRUCTIONS)

IMMEDIATE TO REGISTER

	IMMEDIATE		
MNEMONIC	BYTE	WORD	
LD B,#bval	4 0	6	
LD X,#xval	4 0	5 0	
LD K,#kval	4 0	6 0	
LD BK,#bval,#kval	6 0	10 0	
ADDS A,#value	4 0		

AUTO-INCREMENT/AUTO-DECREMENT

X REGISTER	
MNEMONIC	[X+] [X-]
LD A,	4 1
Х А,	5 2

B REGISTER WITH CONDITIONAL SKIP	
MNEMONIC	[B+] [B-]
LDS A,	4 1
XS A,	5 2

STACK POINTER	
MNEMONIC	DIRECT, BASE, WORD
POP	8 2/1
PUSH	8 2/1

	TIONS WITH IT OPERANDS
CLR A	2 0
INC A	2 0
DEC A	2 0
COMP A	2 0
SWAP A	6 0
RRC A	3 0
RLC A	3 0
SHR A	3 0
SHL A	3 0
IFC	3 0
IFNC	3 0
SC	2 0
RC	2 0

	INSTRUCTIONS THAT TRANSFER CONTROL		
JSRP	11 3		
JSR	8 1		
JSRL	10 1		
JР	4 0		
JMP	5 0		
JMPL	8 0		
Л	7 1		
ЛDW	7 1		
RET	6 1		
RETI	6 1		
RETSK	6 1		

C.13 HPC OPCODE MAP

LSB / MSB \rightarrow

	0	1	2	3	4	5	6	7
0	CLR A COMP A	IFBIT 0	JSRP 0 JSRP 1	JSR + JSR +	JP +1* JP +2	JP +17 JP +18	JP 0 JP -1	JP -16 JP -17
1 2	SC SC	IFBIT 2	JSRP 2	JSR +	JP +3	JP +19	JP -2	JP -17 JP -18
3	RC	IFBIT 3	JSRP 3	JSR +	JP +4	JP +20	JP -3	JP -19
4	INC A	IFBIT 4	JSRP 4	JSR -	JP +5	JP +21	JP -4	JP -20
5	DEC A	IFBIT 5	JSRP 5	JSR -	JP +6	JP +22	JP -5	JP -21
6	IFNC	IFBIT 6	JSRP 6	JSR -	JP +7	JP +23	JP -6	JP -22
7	IFC	IFBIT 7	JSRP 7	JSR -	JP +8	JP +24	JP -7	JP -23
8	SBIT O	RBIT 0	JSRP 8	RBIT X	JP +9	JP +25	JP -8	JP -24
9	SBIT 1	RBIT 1	JSRP 9	SBIT X	JP +10	JP +26	JP −9	JP -25
Α	SBIT 2	RBIT 2	JSRP 10	IFBIT X	JP +11	JP +27	JP -10	JP -26
В	SBIT 3	RBIT 3	JSRP 11	SWAP A	JP +12	JP +28	JР -11	JP -27
C	SBIT 4	RBIT 4	JSRP 12	RET	JP +13	JP +29	JP -12	JP -28
D	SBIT 5	RBIT 5	JSRP 13	RETSK	JP +14	JP +30	JP -13	JP -29
E	SBIT 6	RBIT 6	JSRP 14	RETI	JP +15	JP +31	JP -14	JP -30
F	SBIT 7	RBIT 7	JSRP 15	POP	JP +16	JP +32	JP -15	JP -31
	8	9	A	В	C	D	E	F
0	Dir-Dir	LD A,i	Dir-Dir	LD A,ii	LDS [B+].b	LD [X+],b	LDS [B+].w	LD [X+]w
1	Dir-Dir	LD A,i LD K,i	Dir-Dir	LD K,ii	XS [B+].b	X [X+].b	XS [B+].w	X [X+] w
1 2	Dir-Dir Imm-Dir	LD A,i LD K,i LD B,i		LD K,ii LD B,ii	XS [B+].b LDS [B-].b	X [X+],b LD [X-],b	XS [B+].w LDS [B-].w	X [X+].w LD [X-].w
1 2 3	Dir-Dir Imm-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i	Dir-Dir Index —	LD K,ii LD B,ii LD X,ii	XS [B+].b LDS [B-].b XS [B-].b	X [X+],b LD [X-],b X M[X-],b	XS [B+],w LDS [B-],w XS [B-],w	X [X+].w LD [X-].w X [X-].w
1 2 3 4	Dir-Dir Imm-Dir Imm-Dir Dir-Dir	LD A,i LD K,i LD B,i LD X,i JMP+	Dir-Dir Index — Dir-Dir	LD K,ii LD B,ii LD X,ii JMPL	XS [B+].b LDS [B-].b XS [B-].b LD [B].b	X [X+],b LD [X-],b X M[X-],b LD [X],b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w	X [X+] w LD [X-] w X [X-] w LD [X] w
1 2 3 4 5	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP-	Dir-Dir Index — Dir-Dir Dir-Dir	LD K,ii LD B,ii LD X,ii JMPL JSRL	XS [B+],b LDS [B-],b XS [B-],b LD [B],b X [B],b	X [X+],b LD [X-],b X M[X-],b LD [X],b X [X],b	XS [B+],w LDS [B-],w XS [B-],w LD [B],w X [B],w	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w
1 2 3 4 5 6	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct	Dir-Dir Index — Dir-Dir Dir-Dir Index	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct	XS [B+],b LDS [B-],b XS [B-],b LD [B],b X [B],b ST [B],b	X [X+],b LD [X-],b X M[X-],b LD [X],b X [X],b ST [X],b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w ST [X].w
1 2 3 4 5	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP-	Dir-Dir Index — Dir-Dir Dir-Dir	LD K,ii LD B,ii LD X,ii JMPL JSRL	XS [B+],b LDS [B-],b XS [B-],b LD [B],b X [B],b	X [X+],b LD [X-],b X M[X-],b LD [X],b X [X],b	XS [B+],w LDS [B-],w XS [B-],w LD [B],w X [B],w	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w
1 2 3 4 5 6 7	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct LD bd,i	Dir-Dir Index — Dir-Dir Dir-Dir Index LD BK,ii	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct LD wd,ii	XS [B+].b LDS [B-].b XS [B-].b LD [B].b X [B].b ST [B].b SHR A	X [X+],b LD [X-],b X M[X-],b LD [X],b X [X],b ST [X],b RRC A	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w SHL A	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w ST [X].w RLC A
1 2 3 4 5 6 7	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct LD bd,i ADD A,i AND A,i	Dir-Dir Index — Dir-Dir Dir-Dir Index LD BK,ii LD A,wd INC wd	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct LD wd,ii ADD A,ii AND A,ii	XS [B+].b LDS [B-].b XS [B-].b LD [B].b X [B].b ST [B].b SHR A ADC A,b DADC A,b	X [X+],b LD [X-],b X M[X-],b LD [X],b X [X],b ST [X],b RRC A ADD A,b AND A,b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w SHL A ADC A,w DADC A,w	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w ST [X].w RLC A ADD A,w AND A,w
1 2 3 4 5 6 7	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir Imm-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct LD bd,i ADD A,i AND A,i OR A,i	Dir-Dir Index — Dir-Dir Dir-Dir Index LD BK,ii LD A,wd INC wd DECSZ wd	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct LD wd,ii ADD A,ii AND A,ii OR A,ii	XS [B+].b LDS [B-].b XS [B-].b LD [B].b X [B].b ST [B].b SHR A ADC A,b DADC A,b DSUBC A,b	X [X+],b LD [X-],b X M[X-],b LD [X],b X [X],b ST [X],b RRC A ADD A,b AND A,b OR A,b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w SHL A ADC A,w DADC A,w DSUB A,w	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w ST [X].w RLC A ADD A,w AND A,w OR A,w
1 2 3 4 5 6 7 8 9 A B	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir Imm-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct LD bd,i ADD A,i AND A,i OR A,i XOR A,i	Dir-Dir Index — Dir-Dir Dir-Dir Index LD BK,ii LD A,wd INC wd DECSZ wd ST A,wd†	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct LD wd,ii ADD A,ii AND A,ii OR A,ii XOR A,ii	XS [B+].b LDS [B-].b XS [B-].b LD [B].b X [B].b ST [B].b SHR A ADC A,b DADC A,b DSUBC A,b SUBC A,b	X [X+],b LD [X-],b X M[X-],b LD [X],b ST [X],b RRC A ADD A,b AND A,b OR A,b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w SHL A ADC A,w DADC A,w DSUB A,w SUBC A,w	X [X+].w LD [X-].w X [X-].w LD [X].w ST [X].w ST [X].w RLC A ADD A,w AND A,w OR A,w XOR A,w
1 2 3 4 5 6 7 8 9 A B	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir Imm-Dir Imm-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct LD bd,i ADD A,i AND A,i OR A,i IFEQ A,i	Dir-Dir Index Dir-Dir Dir-Dir Index LD BK,ii LD A,wd INC wd DECSZ wd ST A,wd† LD wd,wd	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct LD wd,ii ADD A,ii AND A,ii OR A,ii IFEQ A,ii	XS [B+],b LDS [B-],b XS [B-],b LD [B],b X [B],b ST [B],b SHR A ADC A,b DADC A,b DSUBC A,b SUBC A,b JID	X [X+],b LD [X-],b X M[X-],b LD [X],b ST [X],b RRC A ADD A,b AND A,b OR A,b IFEQ A,b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w SHL A ADC A,w DADC A,w DSUB A,w SUBC A,w JIDW	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w ST [X].w RLC A ADD A,w AND A,w OR A,w IFEQ A,w
1 2 3 4 5 6 7 8 9 A B C D	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir Imm-Dir Imm-Dir LD A,bd INC bd DECSZ bd ST A,bd† LD bd,bd LD BK,i	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct LD bd,i ADD A,i AND A,i OR A,i IFEQ A,i IFGT A,i	Dir-Dir Index Dir-Dir Dir-Dir Index LD BK,ii LD A,wd INC wd DECSZ wd ST A,wd† LD wd,wd Indirect	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct LD wd,ii ADD A,ii AND A,ii OR A,ii IFEQ A,ii IFGT A,ii	XS [B+].b LDS [B-].b XS [B-].b LD [B].b X [B].b ST [B].b SHR A ADC A,b DADC A,b DSUBC A,b SUBC A,b	X [X+],b LD [X-],b X M[X-],b LD [X],b X [X],b ST [X],b RRC A ADD A,b AND A,b OR A,b IFEQ A,b IFGT A,b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w SHL A ADC A,w DADC A,w DSUB A,w SUBC A,w	X [X+].w LD [X-].w X [X-].w LD [X].w ST [X].w ST [X].w RLC A ADD A,w AND A,w OR A,w IFEQ A,w IFEQ A,w
1 2 3 4 5 6 7 8 9 A B	Dir-Dir Imm-Dir Imm-Dir Dir-Dir Dir-Dir Imm-Dir Imm-Dir Imm-Dir Imm-Dir	LD A,i LD K,i LD B,i LD X,i JMP+ JMP- Direct LD bd,i ADD A,i AND A,i OR A,i IFEQ A,i	Dir-Dir Index Dir-Dir Dir-Dir Index LD BK,ii LD A,wd INC wd DECSZ wd ST A,wd† LD wd,wd	LD K,ii LD B,ii LD X,ii JMPL JSRL Direct LD wd,ii ADD A,ii AND A,ii OR A,ii IFEQ A,ii	XS [B+],b LDS [B-],b XS [B-],b LD [B],b X [B],b ST [B],b SHR A ADC A,b DADC A,b DSUBC A,b SUBC A,b JID	X [X+],b LD [X-],b X M[X-],b LD [X],b ST [X],b RRC A ADD A,b AND A,b OR A,b IFEQ A,b	XS [B+].w LDS [B-].w XS [B-].w LD [B].w X [B].w ST [B].w SHL A ADC A,w DADC A,w DSUB A,w SUBC A,w JIDW	X [X+].w LD [X-].w X [X-].w LD [X].w X [X].w ST [X].w RLC A ADD A,w AND A,w OR A,w IFEQ A,w

- means opcode is reserved for future use.

b = byte of memory

w = word of memory

bd = direct byte of memory

wd = direct word of memory

i = 8-bit immediate value ii = 16-bit in

ii = 16-bit immediate value

Dir-Dir, Imm-Dir, Index, Direct, Indirect and XIndirect are all

Addressing Mode directives. See Section C.2.

NOTES: * NOP is the same as JP +1 and has the same opcode.

† These opcodes are LD if prefixed by Dir-Dir or Imm-Dir directive.

Appendix D

BASIC ASSEMBLY LANGUAGE SYNTAX

D.1 ASSEMBLER SYNTAX

At the beginning of the program, the user can define variables and constants. For example:

VAR = 36:WORD

defines a word variable called VAR at address 36. Having done this, the programmer can refer to this variable by name. In instructions, the assembler uses the syntax expression.W for words and expression.B for bytes of memory.

Examples of the various addressing mode formats are as follows:

Register Indirect

LD A,[B]B	Loads A with the byte of memory at location addressed by contents of register B.	the
LD A,[B]W	Loads A with the word of memory at location addressed by contents of register B.	the

Immediate

LD	A,#22	Loads A with immediate value of decimal 22.
LD	A,#VAR	Loads A with immediate value of 36.

Direct

LD A,22.W	Loads A with word of memory at location 22.
LD A,22.B	Loads A with byte of memory at location 22.
LD A,VAR	Loads A with word of memory at location 36.

Indirect

LD A, [VAR].W Loads A with the word of memory at the location addressed by the contents of VAR (the word of memory at location 36).

LD [68.W].B Loads A with the byte of memory at the location addressed by the contents of the word of memory at location 68.

Indexed

LD A, [VAR+6].W Loads A with the word at the memory location addressed by the contents of VAR plus 6.

To Direct Memory

LD 10.B,69.B Copies the contents of memory at location 69 to memory location

10.

LD VAR,#25 Stores the value 25 to VAR (word at 36). (The byte value 25 is

zero-extended to 16 bits first.)

Register Indirect (auto increment & decrement)

LD A, [X+].B Loads A with byte of memory at location addressed by X, then

increments X by 1.

LD A, [X-].W Loads A with word of memory at location addressed by X, then

decrements X by 2.

Bit Addressing (Static)

SBIT 3, [B]B Sets bit 3 of the byte of memory addressed by the contents of

register B.

RBIT 7,10.B Clears bit 7 of the byte of memory at address 10.

Bit Addressing (Dynamic)

IFBIT X, [B].B Tests the bit at the position given by the least-significant 3 bits of

register X; the byte's address is B + (X/8).

Appendix E

ADDRESSING MODES AND BINARY FORMATS

E.1 INTRODUCTION

As stated in Chapter 4, the general addressing modes of the HPC fall into two categories: those that are available to all instructions that use general modes (One-Address modes), and those that are supported only by the Two-Address subset of these instructions (Two-Address modes). This chapter defines each addressing mode in full detail, and also defines the binary construction of instructions based on their addressing modes.

The One-Address modes are:

- Register B Indirect (Section E.3.1)
- Register X Indirect (Section E.3.2)
- Direct (Section E.3.3)
- Indirect (Section E.3.4)
- Indexed (Section E.3.5)

The Two-Address modes are:

- Immediate to Accumulator (Section E.4.1)
- Direct Memory to Direct Memory (Section E.4.2)
- Immediate to Direct Memory (Section E.4.3)

E.2 ENCODING SCHEMES

Each instruction in the HPC instruction set has a characteristic 8-bit Operation Code ("opcode"). Depending on the addressing mode selected, the opcode byte may constitute the entire instruction, or the instruction may expand to a maximum of 6 bytes.

An HPC instruction falls into one of the three formats shown below.

Opcode

The Opcode byte stands alone when it encodes the Register B Indirect mode, or when the instruction involves a specialized mode or no mode at all.

Addressing Directive Byte	Addressing Value(s) [as needed]	Opcode
---------------------------------	---------------------------------------	--------

This scheme is used to select modes other than Register B Indirect for the One-Address and Two-Address classes of instructions. An Addressing Directive byte is used as a prefix, followed by any addressing values required by the mode selected by the prefix. The Opcode byte is the last byte of the instruction.

Opcode	Addressing Value(s) [as needed]
--------	---------------------------------------

This scheme is used in instructions where a general addressing mode does not apply, but where addressing values are still needed; for example, in instructions that jump. This also appears as an "optimized" form of a general addressing mode for a small number of instructions, where the specific mode is encoded within the opcode byte itself instead of by an Addressing Directive prefix. For example, the Load instruction form "LD 20.B, 30.B" has this format because both operands are in the Basepage address range and they are the same size.

E.3 ONE-ADDRESS GENERAL MODES

In these modes, there is one general operand. When used in a One-Address instruction, this operand is usually the only value involved, and it serves as a destination. When used in a Two-Address instruction, there are two operands: the Accumulator is the destination and the general operand is the source, and both must be specified in assembly code. The entire Accumulator is always modified in these cases; the source operand is zero-extended to 16 bits internally before use. In instructions that generate a carry, the source operand determines the position from which the C bit is loaded: selecting a byte source causes the carry to be taken out of the low-order byte of the result, and selecting a word source causes the carry to be taken from the high-order end of the 16-bit result.

E.3.1 Register B Indirect

Syntax:

[B].B or [B].W

In this addressing mode, the operand is the memory location addressed by the contents of the B register.

Opcode

Example:

Machine (Hex)	Mnemonic	Comments
C4	LD A,[B]B	; Accumulator is loaded with byte of memory ; addressed by the B register.

An alternate encoding scheme for this mode is used for the DECSZ and INC instructions. This uses the Indirect mode, and specifies the B register as the Pointer Word using its memory-mapped address (OOCC Hex).

INDIRECT Directive (AD Hex)	CC Hex (Address of B)	Opcode
-----------------------------------	--------------------------	--------

Machine (Hex)	Mnemonic		Comments
AD CC A9	INC [B].W		; The word in memory addressed by the ; B register is incremented.

E.3.2 Register X Indirect

Syntax: [X]B or [X]W

Register X contains the address of the operand. Instructions using Register X Indirect addressing are generally two bytes long, using an addressing mode directive byte as a prefix to select the addressing mode.

X INDIRECT Directive (8F Hex)	Opcode
-------------------------------------	--------

Example:

Machine (Hex)	Mnemonic	Comments
8F C8	ADC A,[X],B	; Byte of memory addressed by the X register is ; added with carry to the Accumulator

In the instructions LD, ST and X, which perform only data movement, a more compact encoding scheme is used. An opcode is dedicated to this mode exclusively, so that no addressing directive is needed, and the instruction occupies only one byte.

Opcode

Machine (Hex)	Mnemonic	Comments
F6	ST A,[X].W	; Accumulator is stored in word of memory ; addressed by the X register.

E.3.3 Direct

Syntax: address.B or address.W

The instruction contains an 8-bit or 16-bit address field that directly specifies the address of the operand. In the general case, the instruction consists of an Addressing Directive byte, which also encodes whether the operand is at a Basepage address. This is followed by an address; 8 bits long for a Basepage address, and 16 bits otherwise. The opcode itself appears as the last byte.

DIRECT Directive (96 Hex)	Address (8 Bits)	Opcode
---------------------------------	---------------------	--------

Example:

Machine (Hex)	Mnemonic	Comments
96 7F D8	ADD A, H'7F.B	; The byte of memory at address 007F; is added to the Accumulator.

Machine (Hex)	Mnemonic	Comments
B6 CF 94 B6	ADD A, H'CF9	The byte operand at address; CF94 is added to the Accumulator.

When the data movement instructions LD, ST or X are used with a Basepage operand, a special form is used with a dedicated opcode. For non-Basepage operands, the form above is still used.

Opcode Address (8 Bits)

Machine (Hex)	Mnemonic	Comments	
AE FC	X A, 252.W	; The contents of the Accumulator are exchanged ; with the word at address 252 (00FC Hex).	

E.3.4 Indirect

Syntax: [address.W].B or [address.W].W

The instruction specifies the 8-bit address of a Pointer Word (in the Basepage addressing range). The addressed Pointer Word contains the 16-bit address of the operand. The operand itself may be a byte or a word in size. Note, however, that the 8-bit address specified must always be an even number, because it is the address of a word.

INDIRECT Add Directive (AD Hex) (8	Vord Opcode
------------------------------------	-------------

Machine (Hex)	Mnemonic	Comments
AD FC 88	LD A, [252.W] B	; The contents of the word at location 252; (Hex 00FC) form an address. The byte at this; address is loaded into the Accumulator.
AD 78 F8	ADD A,[H78.W].W	; The contents of the word at location 0078; (Hex) form an address. The word at this; address is added to the Accumulator.

E.3.5 Indexed

Syntax: disp[address.W].B or disp[address.W].W

The instruction contains an 8-bit address field and also an 8-bit or 16-bit displacement field. The contents of the Pointer Word addressed by the address field is added to the displacement, yielding the address of the operand. The actual operand finally addressed may be a byte or a word. Note, however, that the 8-bit address must be an even number always, because it points to a word.

INDEXED Directive (A2 Hex)	Disp Addr 8 Bits) (8 Bi	Oncode
----------------------------	----------------------------	--------

Example:

Machine (Hex)	Mnemonic	Comments
A2 3E 2D FA	OR A, H'3E [H'2D.W].W	; The contents of the word at 002D are ; added to the displacement H'3E thus ; forming an address. The word at ; this address is fetched and ORed ; with the Accumulator.

INDEXED Directive (A6 Hex)	Disp (MSB)	Disp (LSB)	Address (8 Bits)	Opcode
----------------------------	---------------	---------------	---------------------	--------

Machine (Hex)		Mnemonic	Comments
A6 1F FF 36 8B	ST	A, H'1FFF [H'36.W]. B	; The contents of the word at 0036 is ; added to the displacement 01FFF thus ; forming an address. The Accumulator ; is stored to the byte at this address.

E.4 TWO-ADDRESS GENERAL MODES

The following addressing modes are available in the Two-Address class of instructions.

E.4.1 Immediate

Syntax: A, #value

The specified immediate value is the source operand, and the Accumulator is specified as the destination. This addressing mode does not have an Addressing Directive associated with it; instead, the Immediate form of each instruction is encoded by an additional opcode. The instruction has an 8-bit opcode, followed by an 8-bit or 16-bit field containing the immediate value. 8-bit values are zero-extended to 16 bits internally before use, and the entire Accumulator is affected. In instructions that generate a carry, the C bit is loaded from the high-order end of the full 16-bit result. The only exception to this is the ADDS instruction, which generates the carry from the high-order end of the low-order byte of the result.

Opcode Value (8 Bits)	Opcode	,
-----------------------	--------	---

Example:

Machine (Hex)	Mnemonic	Comments	
9B FC	XOR A, #252	; 252 (00FC Hex) is Exclusive ORed; into the Accumulator.	

Opcode	Immediate (MSB)	Immediate (LSB)
--------	--------------------	--------------------

Machine (Hex)	Mnemonic	Comments
BF 0C 84	DIV A, #H'0C84	; The Accumulator is divided by 0C84 (Hex).

A group of instructions (ADC, SUBC, DADC and DSUBC) are encoded differently from the above. Instead of using a dedicated opcode to express Immediate mode, the Immediate-Direct addressing mode is used, specifying the Accumulator as the destination through its memory-mapped address (00C8 Hex). This form uses an Addressing Directive.

IMM-DIR Directive (82 Hex)	Immediate Value	C8 Hex (Address of A)	Opcode
----------------------------------	--------------------	--------------------------	--------

Example:

Machine (Hex)	Mnemonic	Comments
82 20 C8 E8	ADC A, #32	; 32 (0020 Hex) is added with carry to the Accumulator.

IMM-DIR Directive (86 Hex)	mmediate (MSB)	Immediate (LSB)	C8 Hex (Address of A)	Opcode
----------------------------	-------------------	--------------------	--------------------------	--------

Example:

Machine (Hex)	Mnemonic	Comments
86 14 19 C8 EA	DSUBC A, #H'1419	; 1419 (Hex) is decimally subtracted ; with carry from the Accumulator.

E.4.2 Direct Memory To Direct Memory

In the Direct Memory to Direct Memory Mode, there are two operands. The destination may be specified as a memory location or as any register (by using that register's memory-mapped address). The source operand may be a memory location. The source operand is allowed to differ in size from the destination operand; it is zero-extended to 16 bits if necessary before use. In instructions that generate a carry, the destination operand determines the position from which the C bit is loaded: selecting a byte destination causes the carry to be taken out of the low-order byte of the result, and selecting a word destination causes the carry to be taken from the high-order end of the 16-bit result.

Syntax: destaddr.destlen, sourceaddr.sourcelen

where destaddr and sourceaddr are the addresses of the destination and source operands, respectively, and destlen and sourcelen are their length suffixes: "B" for Byte, and "W" for Word.

The instruction contains an 8-bit or 16-bit address field for the source and, independently, an 8-bit or 16-bit address field for the destination. Both the source and the destination may be byte or word size, also independently. It is not recommended however, to use a byte size destination with a word size source.

DIR-DIR Directive (1 of 8)	Source Address	Dest Address	Opcode

There are eight possible addressing directives that apply here, depending on the size of the source operand, the size of the source address field and the size of the destination address field. The size of the destination operand itself is encoded within the Opcode byte. Possible Addressing Directive bytes are:

Directive	Source Size	Source Address	Dest. Address	Size of Instr.
80	Byte	8-Bit	8-Bit	4 Bytes
A0	Word	8-Bit	8-Bit	4 Bytes
84	Byte	16-Bit	8-Bit	5 Bytes
A4	Word	16-Bit	8-Bit	5 Bytes
81	Byte	8-Bit	16-Bit	5 Bytes
A1	Word	8-Bit	16-Bit	5 Bytes
85	Byte	16-Bit	16-Bit	6 Bytes
A5	Word	16-Bit	16-Bit	6 Bytes

When the source address or the destination address is 16 bits long, its most-significant byte appears first.

Examples:

Machine (Hex)	Mnemonic		Comments
80 FE FC F8	ADD	252 .W, 254 .B	; The byte operand at memory location ; 254 (FE) is added to the word ; destination at memory location 252 (FC).
A0 FE FC F8	ADD	252 .W, 254 .W	; The word operand at memory location 254 (FE); is added to the word at memory location; 252 (FC) where the result is stored.
84 01 00 FC 8B	LD	252 . B, 256 . B	; The byte operand at memory location ; 256 (0100) is loaded into the byte size ; destination at memory location 252 (FC).
A4 01 00 FC AB	LD	252 .W, 256 .W	; The word operand at location 256 is ; moved into the word size destination ; at location 252.
81 FC 01 00 AB	LD	256 .W, 252 .B	; The byte operand at location 252 is ; moved into the word size destination ; at location 256.
A1 FC 01 00 8B	LD	256 .W, 252 .W	; The word operand at location 252 is ; moved into the word size destination ; at location 256.
85 01 02 01 00 8B	LD	256 . B, 258 . B	; The byte operand at location 258 is ; moved into the byte size destination ; at location 256.
A5 01 02 01 00 AB	LD	256 .W, 258 .W	; The word operand at location 258 is ; moved into the word size destination ; at location 256.

NOTE: The addresses used above are for illustration only; they do not correspond to valid on-chip memory addresses.

The LD instruction uses the special encoding case below when both operands are in the Basepage range (8-bit addresses) and they are both the same size. Otherwise, the form above still applies.

Opcode	Source Address (8 Bits)	Dest. Address (8 Bits)
--------	-------------------------------	------------------------------

Machine (Hex)	Mnemonic	Comments
8C FE FC	LD 252.B, 254.B	; The byte operand at location 254 (FE) is ; loaded into the byte at memory location 252 (FC).
AC FE FC	LD 252.W, 254.W	; The word operand at location 254 (FE) is ; loaded into the word at memory location 252 (FC).

E.4.3 Immediate To Direct Memory

In the Immediate to Direct Memory Mode, there are two operands. The destination may be specified as a memory location or as any register (by using that register's memory-mapped address). The source operand is an immediate value. The source operand is allowed to differ in size from the destination operand; it is zero-extended to 16 bits if necessary before use. In instructions that generate a carry, the destination operand determines the position from which the C bit is loaded: selecting a byte destination causes the carry to be taken out of the low-order byte of the result, and selecting a word destination causes the carry to be taken from the high-order end of the 16-bit result.

Syntax:

destaddr.B,#value or destaddr.W,#value

where "destaddr" is the address of the destination operand, and "value" is the immediate source value.

This form allows a memory location to be modified directly, using an immediate value as the source operand. The instruction contains an 8-bit or 16-bit immediate operand and an 8-bit or 16-bit field specifying the destination address. The size of the destination may be byte or word. It is not recommended however, to use an immediate value greater than 255 when the destination is byte size.

IMM-DIR Directive (1 of 4)	Immediate Value	Dest Address	Opcode
----------------------------------	--------------------	-----------------	--------

There are four possible addressing directives that apply here, depending on the size of the immediate value and the size of the destination address field. The size of the destination operand itself is encoded within the Opcode byte. Possible Addressing Directive bytes are:

Directive	Immediate Size	Dest. Address	Size of Instr.
82	8-Bit	8-Bit	4 Bytes
83	8-Bit	16-Bit	5 Bytes
86	16-Bit	8-Bit	5 Bytes
87	16-Bit	16-Bit	6 Bytes

Examples:

Machine (Hex)	М	nemonic	Comments
82 FE FC C8	ADC	252 . B,#254	; The 8-bit immediate value 254 (FE) is ; added with carry to the byte ; size operand at memory location 252 ; (FC) where the result is stored.
82 FE FC E8	ADC	252.W, # 254	; The 8-bit immediate value 254 (FE) is ; added with carry to the word ; size operand at memory location 252 ; (FC) where the result is stored.
86 01 00 FC EB	SUBC	252.W,#256	; The byte operand at memory location ; 256 (FC) is subtracted with carry from ; the byte operand at location 256 (0100) ; where it is then stored.
83 FC 01 00 8B	LD	256.B,#252	; The 8-bit immediate operand 252 is ; loaded into the byte size destination ; at memory location 256.
83 FC 01 00 AB	LD	256 .W,#252	; The 8-bit immediate operand 252 is ; loaded into the word size destination ; at memory location 256.
87 01 02 01 00 AB	LD	256 .W,#258	; The immediate operand 258 (0102) is ; moved into the word size destination ; at location 256 (0100).

NOTE: The addresses used above are for illustration only; they do not correspond to valid on-chip memory addresses.

The LD instruction has two special encodings, shown below, which are used when the destination operand is in the Basepage range. (When this is not the case, the encoding scheme above applies.) The immediate value may be either a byte or a word, as selected by the opcode.

LD Opcode (97 Hex)	Immediate Byte	Dest Address (8 Bits)
--------------------------	-------------------	-----------------------------

Example:

Machine (Hex)	Mnemonic	Comments
97 FE FC	LD 252.B,#254	; The 8-bit immediate value 254 (FE) is ; loaded into the byte of memory at ; location 252 (FC).

LD	Immediate	Immediate	Dest
Opcode	Value	Value	Address
(B7 Hex)	(MSB)	(LSB)	(8 Bits)

Example:

Machine (Hex)	Mnemonic	Comments
B7 01 00 FC	LD 252.W,#256	; The 16-bit immediate operand 256 (0100); is moved into the word of memory at; location 252 (FC).

Appendix F

REVISION TO ASSEMBLER SYNTAX

F.1 INTRODUCTION

The assembler syntax used in this manual is that of the current ASMHPC assembler, which has been revised for use with the CCHPC C compiler. Its output is relocatable rather than absolute, and partly because of this there are a significant number of differences in syntax between this assembler and the older preliminary assembler. Some of the changes, which involve instruction mnemonics and the syntax of addressing mode expressions, are covered here. A translator program is in development at this writing, which will convert programs written in the old syntax to the new syntax.

F.2 ADDRESSING MODE DIFFERENCES

Assuming that N is a constant of value 2, and WRD is a word variable in the Basepage space, the following differences in addressing mode expressions appear:

MODE	OLD SYNTAX	NEW SYNTAX
B Indirect	LD A, M(B) LD A, W(B)	LD A,[B].B LD A,[B].W
X Indirect	LD A, M(X) LD A, W(X)	LD A,[X].B LD A,[X].W
Direct	LD A, M(10) LD A, W(10)	LD A, 10.B LD A, 10.W
Indirect	LD A, M(WRD) LD A, W(WRD)	LD A,[WRD].B LD A,[WRD].W
Indexed	LD A, M(WRD + N) LD A, W(WRD + N)	LD A, N [WRD]. B LD A, N [WRD].W
Immediate	LD A, N	LD A,#N *
DirDir.	LD W(22), M(15)	LD 22.W, 15.B
ImmDir.	LD M(15), N	LD 15.B,#N *
Auto-Increment	LD A, W(X+)	LD A,[X+].W
Auto-Decrement	LDS A, M(B -)	LDS A,[B-].B
Bit	SET M(2).5 RESET M(B).0 IF M(B).X	SBIT 5,2.B RBIT 0,[B].B IFBIT X,[B].B

^{* -} A "#" is now required.

F.3 INSTRUCTION MNEMONICS DIFFERENCES

The following instruction mnemonics have also been changed from the earlier assembler:

OLD		NEW
SET		SBIT
RESET		RBIT
IF		IFBIT
SET	C	SC
RESET	C	RC
IF	C	IFC
IFN	C	IFNC
RETS		RETSK

Appendix G

ROMLESS HPC16003

G.1 INTRODUCTION

The HPC16003 is identical to the HPC16083 except it has no on-chip ROM. The ROMless HPC16003 is intended for use with external memory for program storage.

G.2 HPC16003 RESTRICTIONS

- The EXM pin must be pulled high (logic "1") at all times. The absence of on-chip ROM means that the HPC16003 must be used in the Expanded ROMless 8-bit or Expanded ROMless 16-bit mode. All other operating modes found on the HPC16083 are not valid for the HPC16003.
- The reset vector address must be between F000 and FFFF. Therefore, ROM type memory must be found externally at addresses FFFF:FFFE and lower. For example, a 27C32 EPROM can be used with the HPC16003 and addressed from F000 to FFFF. In this case it makes sense to use F000 as the reset vector address and start the program at this address. If 8, 16, 32 or 64 Kbytes of external memory are used, the reset vector address and start of program must still be an address between F000 and FFFF.
- If external memory is used in the range 0200 to EFFF in addition to the required memory near the top of the address range, the EA bit in the PSW register MUST be set to "1" by the user's program to enable proper operation of the external bus, and to prevent inadvertent Watchdog Outputs. It is recommended that the EA bit be set to "1" as part of initialization at the beginning of the program.
- The illegal address detection feature of the Watchdog logic is not available on the HPC16003. All other features of Watchdog are retained.

TABLE G-1. HPC16003 MEMORY MAP

FFFF:FFF0	Interrupt Vectors in off-chip ROM	
FFEF:FFD0	JSRP Vectors	
	in off-chip ROM	USER MEMORY
FFCF:FFCE		
:	External	
0201:0200	Memory	
01FF:01FE		1
:	On-Chip RAM	USER RAM
01C1:01C0		
0195:0194	Watchdog Register	Watchdog Logic
0192	TOCON Register	
0191:0190	TMMODE Register	
018F:018E	DIVBY Register	
018D:018C	T3 Timer	
018B:018A	R3 Register	
0189:0188	T2 Timer	Timer Block T0:T3
0187:0186	R2 Register	
0185:0184	I2CR Register/R1	
0183:0182	13CR Register/ T1	
0181:0180	I4CR Register	
015F:015E	EICR Register	
015C	EICON Register	
0153:0152	PORTP Register	
0151:0150	PWMODE Register	
014F:014E	R7 Register	
014D:014C	T7 Timer	
014B:014A	R6 Register	
0149:0148	T6 Timer	Timer Block T4:T7
0147:0146	R5 Register	
0145:0144	T5 Timer	
0143:0142	R4 Register	
0141:0140	T4 Timer	
0128	ENUR Register	
0126	TBUF Register	
0124	RBUF Register	UART
0122	ENUI Register	
0120	ENU Register	
for the second s	•	

TABLE G-1. (Cont)

0104	PORTD Input Register	PORT D
0104 00F5:00F4	PORTD Input Register BFUN Register	PORT D
00F3:00F4 00F3:00F2	DIRB Register	PORTS A & B CONTROL
00F1:00F0	DIRA Register / IBUF	
00E6	UPIC register	UPI CONTROL
00E3:00E2	PORTB	
00E1:00E0	PORTA Register / OBUF	PORTS A & B
00DE 00DD:00DC 00D8 00D6 00D4	Microcode ROM Dump Halt Enable Register PORTI Input Register SIO Register IRCD Register	PORT I, CONTROL & INTERRUPT CONTROL REGISTERS
00D2 00D0	IRPD Register ENIR Register	
00CF:00CE 00CD:00CC 00CB:00CA 00C9:00C8 00C7:00C6 00C5:00C4 00C3:00C2 00C0	X Register B Register K Register A Register PC Register SP Register (reserved) PSW Register	HPC CORE REGISTERS
00BF:00BE : 0001:0000	On-Chip RAM	USER RAM

		1
		$\overline{}$
		~,

Appendix H

DIFFERENCES IN THE HPC16043

H.1 INTRODUCTION

The HPC16043 is identical to the HPC16083 except it has a 4-Kbyte on-chip ROM. The 4-Kbyte ROM extends from F000 to FFFF, see Table H-1.

H.2 HPC16043 DIFFERENCES (WITH RESPECT TO THE HPC16083)

• The "on-chip ROM range" of the HPC16043 extends from F000 to FFFF, rather than E000 to FFFF as on the HPC16083. The "on-chip ROM range" affects all operating modes except the Expanded ROMless modes. The function of the operating modes is the same as on the HPC16083 except the address ranges are different because of the smaller ROM of the HPC16043. The memory ranges for the operating modes are shown below:

Single Chip Normal Mode

- on-chip ROM is F000:FFFF

Expanded Normal 8-bit and 16-bit Modes

- on-chip ROM is F000:FFFF
- allowable external memory is 0200:EFFF

Single Chip ROMless 8-bit and 16-bit Modes

- allowable external memory is F000:FFFF

Expanded ROMless 8-bit and 16-bit Modes

- allowable external memory is 0200:FFFF
- The illegal address range of Watchdog is 0200 to EFFF for the HPC16043. Accesses to addresses in this range will cause a Watchdog Output if the HPC16083 is in any of the Single Chip modes (EA bit "0").
- The reset vector address must be between F000 and FFFF for the HPC16043.

TABLE H-1. MEMORY MAP OF HPC16043

FFFF:FFF0	Interrupt Vectors	
	in on-chip ROM	
FFEF:FFD0	JSRP Vectors	
TETATE	in on-chip ROM	USER MEMORY
FFCF:FFCE	an our ourp nour	
	On Chin DOM	
F001:F000	On-Chip ROM	
EFFFEFFE	F	
0201:0200	External Expansion	
	Memory	
01FF:01FE		
	On-Chip RAM	USER RAM
01C1:01C0		
0195:0194	Watchdog Register	Watchdog Logic
0192	TOCON Register	
0191:0190	TMMODE Register	
018F:018E	DIVBY Register	
018D:018C	T3 Timer	
018B:018A	R3 Register	
0189:0188	T2 Timer	Timer Block T0:T3
0187:0186	R2 Register	
0185:0184	12CR Register/R1	
0183:0182	I3CR Register/ T1	
0181:0180	I4CR Register	
015F:015E	EICR Register	
015C	EICON Register	
0153:0152	PORTP Register	
0151:0150	PWMODE Register	
014F:014E	R7 Register	
014D:014C	T7 Timer	
014B:014A	R6 Register	
0149:0148	T6 Timer	Timer Block T4:T7
0147:0146	R5 Register	
0145:0144	T5 Timer	
0143:0142	R4 Register	
0141:0140	T4 Timer	
0128	ENUR Register	
0126	TBUF Register	
0124	RBUF Register	UART
0122 .	ENUI Register	
0120	ENU Register	
	1	•

TABLE H-1. (Cont)

0104	PORTD Input Register	PORT D
00F5:00F4 00F3:00F2 00F1:00F0	BFUN Register DIRB Register DIRA Register / IBUF	PORTS A & B CONTROL
00E6	UPIC register	UPI CONTROL
00E3:00E2 00E1:00E0	PORTB PORTA Register / OBUF	PORTS A & B
00DE 00DD:00DC 00D8 00D6 00D4 00D2 00D0	Microcode ROM Dump Halt Enable Register PORTI Input Register SIO Register IRCD Register IRPD Register ENIR Register	PORT I, CONTROL & INTERRUPT CONTROL REGISTERS
00CF:00CE 00CD:00CC 00CB:00CA 00C9:00C8 00C7:00C6 00C5:00C4 00C3:00C2 00C0	X Register B Register K Register A Register PC Register SP Register (reserved) PSW Register	HPC CORE REGISTERS
00BF:00BE : 0001:0000	On-Chip RAM	USER RAM

Appendix I

ASCII CHARACTER SET

CHAR.	7-BIT HEX NUMBER	CHAR.	7-BIT HEX NUMBER	CHAR.	7-BIT HEX NUMBER	CHAR.	7-BIT HEX NUMBER
NUL	00	SP	20	@	40		60
SOH	01	!	21	A	41	a	61
STX	02	**	22	В	42	b	62
ETX	03	#	23	С	43	С	63
EOT	04	\$	24	D	44	d	64
ENQ	05	%	25	E	45	e	65
ACK	06	&	26	F	46	f	66
BEL	07	′	27	G	47	g	67
BS	08	(28	Н	48	h	68
HT	09)	29	I	49	i	69
LF	0A	*	2A	J	4A	j	6A
VT	OB	+	2B	K	4B	k	6B
FF	OC	,	2C	L	4C	1	6C
CR	OD	-	2D	M	4D	m	ഇ
SO	0E		2E	N	4E	n	6 E
SI	OF	/	2F	0	4 F	0	6 F
DLE	10	0	30	P	50	p	70
DC1	11	1	31	Q	51	q	71
DC2	12	2	32	R	52	r	72
DC3	13	3	33	S	53	s	73
DC4	14	4	34	T	54	t	74
NAK	15	5	3.5	U	55	∥ u	75
SYN	16	6	36	V	56	v	76
ETB	17	7	37	W	57	w	77
CAN	18	8	38	X	58	x	78
EM	19	9	39	Y	59	У	79
SUB	1A	:	3A	Z	5A	Z	7A
ESC	1B	;	3B	[5B	{	7B
FS	1C	<	3C	\ \ \	5C	1	7C
GS	1D	=	3D]]	5D	 	7 D
RS	1E	>	3E	1	5E	~	7E
US	1F	?	3F	—	5F	DEL,	7 F
			_			rubout	

		4
		~
		$\overline{}$
		_
		`

INDEX

A		write cycle; one wait state	10-15
A register	3-1		
Accumulator register (A)	3-1	\mathbf{C}	
ADC	5-5		
ADD	5-8	C bit	3-2
Add short immediate	5-11	Carry bit	2-4, 5-2
Add with carry, ADC	5-5	Carry use	2-2
Address register (B)	3-1	Character frame formats (UART)	18-3
Address register (X)	3-1	Clear accumulator	5-15
Addresses, illegal	10-1, 13-1	Clocking	7-1, 18-7
Addressing bytes	2-2	CLR A	5-15
Addressing modes, general	4-1	COMP A	5-16
B register indirect	4-4	Complement accumulator	5-16
direct	4-5	Conditional instructions	2-6
direct-direct	4-6	Control registers	8-1
immediate	4-6	Crystal oscillator connections (typical)	7-2
immediate-direct	4-6		
indexed	4-5	_	
indirect	4-5	D	
X register indirect	4-4		
Addressing modes, specialized		DADC	5-17
B register indirect, auto-inc./dec.	4-7	Data flow between	18-5
B register indirect, conditional skip	4-7	TSFT/RSFT and other registers (UART)	18-5
double register indirect	4-9	Data handling, 8-/16-bit	2-2
X register indirect, auto-inc./dec.	4-7	Data sizes, mixing	. 2-3
ADDS	5-11	DEC A	5-21
AND	5-12	Decimal add with carry	5-17
Architectural features	2-1	Decimal subtract with carry	5-32
Assembler syntax, revision to	F-1	Decrement accumulator	5-21
Attributes	5-1	Decrement and skip if zero	5-22
		DECSZ	5-22
		Dedicated registers	8-1
В		Direct addressing mode	4-5, E-5
		Direct memory to direct memory	E-10
B register	3-1	addressing mode	4-6, E-10
conditional skip	4-7	DIV	5-24
indirect	4-4	DIVD	5-28
indirect with auto-increment/decrement	4-7	Divide	5-24
Block diagram, HPC16083	1-3	Divide double word	5-28
Boundary constant register (K)	3-1	DMA	11-1
Bus configurations	10-1	Double register indirect addressing mode	4-9
Bus functions	10-1	DSUBC	5-32
Bus modes	10-1		
expanded normal 16-bit	10-23	-	
expanded normal 8-bit	10-23	E	
expanded ROMless 16-bit	10-26		
expanded ROMless 8-bit	10-25	EICON (external interrupt config. reg.)	15-6, 16-10
single chip normal	10-21	EICR (external interrupt capture register)	16-7 E-1
single chip ROMless 16-bit	10-25	Encoding schemes, instruction	15-5
single chip ROMless 8-bit	10-24	ENIR (interrupt enable register)	
Bus operation	10-6	Error flags	18-4
Bus timing		Exchange with A, with auto-increment/	5-104 5-104
read cycle; no wait states	10-10	decrement and conditional skip	5-104 5-98
read cycle; one wait state	10-14	Exchange with accumulator	
write cycle, no wait states	10-11	Exclusive or	5-101

The state of the s	10.1	Your and a ship atom	
Expanded mode vs. single chip mode Expanded normal 16-bit bus mode	10-1 10-23	Interrupt arbitration Interrupt control register maps	15-1 15-5
Expanded normal 8-bit bus mode	10-23	Interrupt control registers	15-3 15-3
	10-25	Interrupt latency	15-3 15-5
Expanded ROMless 16-bit bus mode Expanded ROMless 8-bit bus mode	10-25	Interrupt logic, block diagram of	15-3 15-4
expanded kOM162 8-011 ous mode	10-23	Interrupt processing	15-2
		Interrupt-driven UPI application	13-2
F		UPI interface to NSC800	19-10
		UPI interface to Series 32000 system	19-10
Forms Commete managed (IIABT)	18-3	-	15-1, 18-6
Frame formats supported (UART)	10-3	Interrupts	•
		IRCD (interrupt condition register)	15-6
G		IRPD (interrupt pending register)	15-8
, W			
Comment addressing market		J	
General addressing modes	4-4, E-2	•	
one-address	4-4, E-2 4-6, E-9	Л	5-50
two-address	4-0, E-9 7-1	JIDW	5-52
Grounding	1-1	JMP	5-54
		JMPL	<i>5-3</i> 4 5-56
п		JP JP	5-56 5-57
**			5-57 5-59
** 4. 4.	14.1	JSR	5-59 5 -61
Halt mode	14-1	JSRL JSRP	5-62
HALT timing	14-2	*	5-50
using NMI to exit HALT mode	14-2	Jump indirect (byte)	5-50 5-52
Handshaking (UPI port)	19-6	Jump indirect (word)	5-52 5-54
Hardware connections	7-1	Jump relative	5-34 5-56
Host wait state	10.12	Jump relative long	5-57
insertion sequence of events, reading	19-12	Jump relative short	5-59
insertion sequence of events, writing	19-13	Jump to subroutine	
HPC16043	0.3	Jump to subroutine long	5-61 5-62
memory map	G-2	Jump to subroutine short	3-02
HPC16083 block diagram	1-3		
HPC16083	8-2	K	
memory map		R.	
pin description	6-1 6-2	K register	3-1
pinout diagram for	0-2	W teStreet	3-1
		_	
I		L	
Idle mode	14-4	LD	5-64
If carry	5-39	LDS	5-68
If equal	5-40	Load	5-64
-	5-43	Load A, with auto-increment/decrement	5-68
If greater than If not carry	5-46	and conditional skip	5-68
I not carry	5-36	Logical AND	5-12
IFC	5-39	Logical or	5-74
IFEO	5-40	mogram or	•
IFGT	5-43		
IFNC	5-46	M	
	13-1	***	
Illegal addresses Immediate addressing mode	4-6, E-9	Memory map	8-1
	E-14	HPC16003	G-1
Immediate to direct memory addressing mode	4-6, E-14	HPC16043	H-1
•	5-47	HPC16083	8-2, A-2
INC INC A	5-49	Von Neumann architecture	2-1
	5-47	Memory organization/availability chart	10-22
Increment Increment accumulator	5-49	MICROWIRE/PLUS	2 ~ 20
Indexed addressing mode	4-5, E-8	interface block diagram	17-3
Indirect addressing mode	4-5, E-7	transfer	17-3
Infinite loops, potentially	13-1	Mixing data sizes	2-3
Initialization	12-1	Mode setup, UPI	19-7
IIII iid liva iivii	14-1	Annual passing and a	 .

			8-1
MULT Multi-la	5-70 5-70	control dedicated	8-1
Multiply	3-70	interrupt control	15-3
		timer section	16-6
N		UART control	18-9
		Reset	7-3
No operation	5-73	Reset bit	5-79
NOP	5-73	Reset carry	5-81
Normal mode vs. ROMless mode	10-3	Reset state	12-1
Notations, instruction description	5-1	RET	5-82
		REU	5-83
		RETSK	5-84 5-83
0		Return from interrupt Return from subroutine	5-83 5-82
Our address sensors mades	4-4, E-2	Return from subroutine and skip	5-84
One-address general modes	C-18	Revision to assembler syntax	F-1
Opcode map OR	5-74	RLC A	5-85
Organization	5-1	ROMiess mode vs. normal mode	10-3
Organization	• -	Rotate accumulator left through carry	5-85
		Rotate accumulator right through carry	5-86
P		RRC A	5-86
PC register	3-2		
Pin description	6-1	S	
Pinout diagram for HPC16083	6-2		
68-pin PCC package	6-2	SBIT	5-87
POP	5-77	SC	5-89
Pop from the stack	5-77	Set bit	5-87
Port A	9-1	Set carry	5-89
structure of	9-2	Shared memory application	11-2
Port B	9-3	Shared memory support	11-1 5-90
generic structure	9-5 9-7	Shift accumulator left	5-90 5-91
pins with bus control roles	5, 9-6, 9-7	Shift accumulator right SHL A	5-90
structure of 9- timer synchronous pins	3, 5-0, 5- 1 9-6	SHR A	5-91
Port D	9-11	Single chip mode vs. expanded mode	10-1
Port I	9.9	Single chip normal mode	10-21
Port P	9-11	Single chip ROMless 16-bit bus mode	10-25
Power	7-1	Single chip ROMless 8-bit bus mode	10-24
Power save modes of operation	14-1	Skipping	2-6
Power supply connections	7-1	SP register	3-2
Power-on reset circuit	7-4	Specialized addressing modes	4-7
Processor status word	8-5	ST	5-92
Program counter register (PC)	3-2	Stack pointer register (SP)	3-2
Programing model	3-1	Status pins ST1 and ST2	10-20 5-92
Programming considerations in 8-bit mode	10-26	Store accumulator Structure of Port A	9-2
PSW	8-5 5-78	Structure of Port B	7-2
PUSH	5-78	generic	9-5
Push onto the stack PWM Timers	5-76	pins with bus control roles	9-7
block diagram T4-T7	16-6	timer synchronous pins	9-6
T4-T7	16-4	Structure of Port D	9-11
14 1/		Structure of Port P	9-12
		SUBC	5-94
R		Subtract with carry	5-94
		SWAP A	5-97
RBIT	5-79	Swap nibbles of accumulator	5 97
RC	5-81	System bus behavior	10-21
Read operation, UPI	19-6	System configuration tables	10-6
READY signal timing	10-17	System configuration	10-5
Register B indirect	E-3	16-bit modes, gating chip selects 16-bit modes, gating write strobe	10-3
Register X indirect addressing mode	E-4	8-bit modes, gatting write strone	10-17
Registers	17-1	O_OIT TROOM	10-7

chart	10-8	X	
single chip mode	10-2		
		X instruction	5-98
		X register	3-1
T		indirect	4-4
-		indirect with auto-increment/decrement	4-7
Test bit	5-36	XOR	<i>5</i> -101
Timer	16-1	XS	5-101
Timer applications	16-15	210	3-104
Timer block diagram	10.10		
timers TO, T1, T8	16-3		
timers T2 and T3	16-5		
Timer configuration	16-14		
Timer operations	16-1		
Timer outputs	16-4		
Timer section registers	16-6		
Timer setup	16-14		
Timer TO	16-1		
Timer T1	16-2		
Timer T8	16-1		
Two-address general modes	4-6, E-9		
I wo-address general modes	4-0, E-9		
υ			
b			
UART	18-1		
block diagram	18-2		
clock sources from DIVBY register	18-8		
control registers	18-9		
interrupt generation	18-6		
operation	18-1		
register bits, detailed description of	18-9		
Universal peripheral interface, (UPI)	19-1		
UPI	19-1		
application inserting WAIT states	19-11		
block diagram	19-2		
control register	19-4		
interrupt-driven application	19-9, 19-10		
mode setup	19-7		
read operation	19-6	•	
simple application	19-8		
timing diagrams	19-3		
write operation	19-5		
v			
¥			
Von Neumann	2-1		
memory-mapped architecture	2-1		
1977			
W			
Wait states	10-13		
requesting	10-13		
Wake-up mode	18-7		
Watchdog logic	13-1		
Word external memories	10-18		
Words	2-2		
Write operation, UPI	19-5		

FOLD, STAPLE, AND MAIL

424410897-001A

READER'S COMMENT FORM

In the interest of improving our documentation, National Semiconductor invites your comments on this manual.

Please restrict your comments to the documentation. Technical Support may be contacted at:

(800) 538-1866 - U.S. non CA

(800) 672-1811 - CA only

(800) 223-3248 - Canada only

Please rate this document according to the following categories. Include your comments below.

	EXCELLENT	GOOD	ADEQUATE	FAIR	POOR
Readability (style)	0	-			
Technical Accuracy					
Fulfills Needs					
Organization	0				
Presentation (format)				a	
Depth of Coverage	0				
Overall Quality					
NAME				DATE	
TITLE					
COMPANY NAME/DE	PARTMENT				
ADDRESS					
CITY			STATE		ZIP
Do you require a respon-	se? □ Yes □ No	PHONI	Ε		
Comments:					
				•	
	· · · · · · · · · · · · · · · · · · ·				



lhintalianianalidadaalaalaalaali

BUSINESS REPLY MAIL

FIRST CLASS

PERMIT NO. 409

SANTA CLARA, CA

POSTAGE WILL BE PAID BY ADDRESSEE

National Semiconductor Corporation
Microcomputer Systems Division
Technical Publications Dept. 8278, M/S 7C261
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-9968

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES



•			



National Semiconductor Corporation

Microcomputer Systems Division

National Semiconductor Corporation

2900 Semiconductor Drive Santa Clara, California 95051 Tel: (408) 721-5000 TWX: (910) 339-9240

National Semiconductor

5955 Airport Road Suite 206 Mississauga, Ontario L4V1R9 Canada Tel: (416) 678-2920 TWX: 610-492-8863

Electronica NSC de Mexico SA

Hegel No. 153-204 Mexico 5 D.F. Mexico Tel: (905) 531-1689, 531-0569, 531-8204

NS Electronics Do Brasil

Telex: 017-73550

Avda Brigadeiro Faria Lima 830 8 Andar

01452 Sao Paulo, Brasil

Telex: 1121008 CABINE SAO PAULO 113193 INSBR BR

National Semiconductor GmbH

Furstenriederstraße Nr. 5 D-8000 München 21 West Germany Tel.: (089) 5 60 12-0 Telex: 522772

National Semiconductor (UK), Ltd.

301 Harpur Centre Horne Lane Bedford MK40 1TR United Kingdom Tel: 0234-47147 Telex: 826 209

National Semiconductor Benelux

Ave. Charles Quint 545 B-1080 Bruxelles Belgium Tel: (02) 4661807 Telex: 61007

National Semiconductor (UK), Ltd.

1, Bianco Lunos Allė DK-1868 Copenhagen V Denmark

Tel: (01) 213211 Telex: 15179

National Semiconductor

Expansion 10000 28, Rue de la Redoute F-92 260 Fontenay-aux-Roses France

Tel: (01) 660-8140 Telex: 250956

National Semiconductor S.p.A.

Via Solferino 19 20121 Milano Italy

Tel: (02) 345-2046/7/8/9

Telex: 332835

National Semiconductor AB

Box 2016 Stensätravägen 4/11 TR S-12702 Skärholmen Sweden Tel: (08) 970190 Telex: 10731

National Semiconductor

Calle Nunez Morgado 9 (Esc. Dcha. 1-A) E-Madrid 16 Spain Tel: (01) 733-2954/733-2958

Telex: 46133

National Semiconductor Switzerland

Alte Winterthurerstrasse 53 Postfach 567 CH-8304 Wallisellen-Zürich Tel: (01) 830-2727

Telex: 59000

National Semiconductor

Pasilanraitio 6C SF-00240 Helsinki 24 Finland

Tel: (90) 14 03 44 Telex: 124854

NS Japan K.K.

POB 4152 Shinjuku Center Building 1-25-1 Nishishinjuku, Shinjuku-ku Tokyo 160, Japan Tel: (03) 349-0811 TWX: 232-2015 NSCJ-J

National Semiconductor Hong Kong, Ltd.

1st Floor, Cheung Kong Electronic Bldg. 4 Hing Yip Street Kwun Tong Kowloon, Hong Kong Tel: 3-899235 Telex: 43866 NSEHK HX

Cable: NATSEMI HX

NS Electronics Pty. Ltd. Cnr. Stud Rd. & Mtn. Highway Bayswater, Victoria 3153 Australia

Tel: 03-729-6333 Telex: AA32096

National Semiconductor PTE, Ltd.

10th Floor Pub Building, Devonshire Wing Somerset Road Singapore 0923 Tel: 652 700047 Telex: NATSEMI RS 21402

National Semiconductor Far East, Ltd. Taiwan Branch

P.O. Box 68-332 Taipei 3rd Floor, Apollo Bidg. No. 218-7 Chung Hsiao E. Rd. Sec. 4 Taipei Taiwan R.O.C. Tel: 7310393-4, 7310465-6 Telex: 22837 NSTW Cable: NSTW TAIPEI

National Semiconductor (HK) Ltd. Korea Liaison Office

6th Floor, Kunwon Bldg No. 2, 1-GA Mookjung-Dong Choong-Ku, Seoul, Korea C.P.O. Box 7941 Seoul

Tel: 267-9473 Telex: K24942